Defence Research and Development Canada | Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

# Concepts of data/information fusion for naval C$^2$ and airborne ISR platforms


*A. Jouan*
*P. Valin*
*É. Bossé*
*DRDC Valcartier*



**Defence R&D Canada – Valcartier**
Technical Report
DRDC Valcartier TR 2004-284
November 2006


Canada

# Concepts of data/information fusion for naval $C^2$ and airborne ISR platforms

A. Jouan
P. Valin
É. Bossé
DRDC Valcartier

**Defence R&D Canada - Valcartier**

Technical Report

DRDC Valcartier TR 2004-284

November 2006

Authors

Alexandre Jouan, Pierre Valin

Approved by

Éloi Bossé

Section Head, Decision Support Systems

Approved for release by

Gilles Bérubé

DRP Chair

This report is the multi-platform extension of 3 previous reports on demonstrations of data/information fusion. It summarizes the deliverables of the “Demonstration of Multi-Platform Data Fusion between Halifax Class Frigate and an Airborne Collaborative Platform” (DFCP) contract # W2207-E1V01 (Dr. Alexandre Jouan, Principal Investigator) under the Scientific Authority of Dr. Éloi Bossé. The airborne platform is the CP-140 Aurora, whose data fusion concepts, algorithms and performance were described in the three DRDC-V reports entitled “Information Fusion Concepts for Airborne Maritime Surveillance and $C^2$ Operations” (TM-2004-281), “Airborne Application of Information Fusion Algorithms to Classification” (TR-2004-282), and “Demonstration of Data/Information Fusion Concepts for Airborne Maritime Surveillance Operations” (TR-2004-283).

# Abstract

The main objective of this report is to analyze methods, techniques, algorithms, rule-based communication protocols and infrastructures needed to establish a Maritime Tactical Picture (MTP). An MTP is, by definition, the combination of the Local Area Picture (LAP) seen by a unit (which may be part of a Task Force) using its own sensors and a Wide Area Picture (WAP) using information, not controlled by the Task Force, provided by high frequency (HF) or ultra high frequency (UHF) radios or satellites. For that purpose a test-bed was developed to test and benchmark all these elements. This test-bed provides the tools needed to study what information should be communicated, as well as when and how, and the impact it has on joint situational awareness.

# Résumé

L’objectif principal de ce rapport est l’analyse des méthodes, techniques, algorithmes, protocoles de communications et infrastructures nécessaires pour établir une situation tactique maritime (STM). Une STM est, par définition, la combinaison d’une situation locale telle que déterminée par une unité (faisant possiblement partie d’une force opérationnelle) en utilisant ses propres capteurs, et une situation globale utilisant de l’information n’étant pas sous le contrôle de ladite force, provenant de radios haute fréquence ou ultra haute fréquence, ou de satellites. Cet objectif a mené à la création d’un banc d’essai pour tester et valider les éléments ci-haut mentionnés. Ce banc d’essai fournit les outils requis pour déterminer la nature de l’information à communiquer, quand et comment elle doit l’être et l’impact qui en résulte sur l’analyse de la situation commune.

This page intentionally left blank.

# Executive summary

The main objective of this report is to analyze methods, techniques, algorithms, rule-based communication protocols and infrastructures needed to establish a Maritime Tactical Picture (MTP). An MTP is, by definition, the combination of the Local Area Picture (LAP) seen by a unit (which may be part of a Task Force) using its own sensors and a Wide Area Picture (WAP) using information, not controlled by the Task Force, provided by high frequency (HF) or ultra high frequency (UHF) radios or satellites. For that purpose a test-bed was developed to test and benchmark all these elements..

This report will focus on:

- Communication and information exchange for a simplified MTP made up of the combination of the LAP as seen by several units (HALIFAX class frigate and aircraft such as the CP-140 Aurora) cooperating in a Task Force

- Algorithmic requirement analysis and algorithm development and enhancement for LAP and WAP establishment, including non-imaging and imaging and information, especially an automated classifier for infrared imagery

- Simulation environments and architectures which are best suited for scenario generation and sensor simulation for multiple collaborating multi-sensor Command and Control Systems (CCSs), and the capability to provide a fusion test-bed as a modular entity to larger High Level Architecture (HLA) operational federations.

This test-bed provides the tools needed to study what information should be communicated, as well as when and how, and the impact it has on joint situational awareness.

Jouan, A., Valin, P., Bossé, É. 2006. Concepts of data/information fusion for naval C2 and airborne ISR plaforms. DRDC Valcartier TR 2004-284. Defence R&D Canada – Valcartier

# Sommaire

L’objectif principal de ce rapport est l’analyse des méthodes, techniques, algorithmes, protocoles de communication et infrastructures nécessaires pour établir une situation tactique maritime (STM). Une STM est, par définition, la combinaison d’une situation locale telle que déterminée par une unité (faisant possiblement partie d’une force opérationnelle) en utilisant ses propres capteurs, et une situation globale utilisant de l’information n’étant pas sous le contrôle de ladite force, provenant de radios haute fréquence ou ultra haute fréquence, ou de satellites. Cet objectif a mené à la création d’un banc d’essai pour tester et valider les éléments ci-haut mentionnés.

Ce rapport se concentrera sur :

- Les communications et l’échange d’information pour une STM simple provenant de la combinaison de situations locales provenant de différentes unités (frégate de type HALIFAX et aéronef tel que le CP-140 Aurora) coopérant dans une Force Opérationnelle

- Une analyse des algorithmes requis et leur développement, ainsi que leurs améliorations, pour l’établissement d’une situation locale et une situation globale incluant des détecteurs et des capteurs imageurs, en particulier des améliorations faites à un classificateur automatique pour l’imagerie infrarouge

- Les environnements de simulation et les architectures qui sont les mieux adaptés pour la génération de scénarios et la simulation de capteurs pour des systèmes de commande et contrôle, et la capacité de produire un banc d’essai modulaire pour la fusion qui pourrait fonctionner dans une fédération obéissant à une architecture de haut niveau (High Level Architecture en anglais, ou HLA).

Ce banc d’essai fournit les outils requis pour déterminer la nature de l’information à communiquer, quand et comment elle doit l’être, et l’impact qui en résulte sur l’analyse de la situation commune.

Jouan, A., Valin, P., Bossé, É. 2006. Concepts of data/information fusion for naval C2 and airborne ISR plaforms. DRDC Valcartier TR 2004-284. R&D pour la défense Canada – Valcartier

# Table of contents

# List of figures

# List of tables

This page intentionally left blank.

# 1. Introduction

As military and intelligence gathering operations evolve towards a global network architecture whose mandate is to maintain continuous and shared awareness through a constant, fast and timely exchange of uncorrupted information, the objective is to push forward the platform-centric data fusion capabilities developed during recent years in the direction of a network-centric data fusion capability where collaborating platforms and sensors are nodes in the global information grid that detect and track events of interest.

This document focuses on the creation of a network-centric data fusion system leveraging the existing capabilities, algorithms and environments developed as part of the projects

- "Demonstration of Image Analysis and Object Recognition Decision Aids for Airborne Surveillance" (Defence Industrial Research contract to Lockheed Martin (LM) Canada No. W2207-8-EC01),

- "Advanced Shipboard Command and Control Technology (ASCACT), Development of an Application Demonstration Platform (ADP) for Multi-Source Data Fusion (MSDF)" (LM Canada contract No. W8477-8-PH01/001-QE), and

- "Study of Real-Time Issues for an Integrated MSDF/STA/RM System for the CPF" (LM Canada contract No. W7701-5-1677/00/B).

This involves the design and implementation of a network architecture ensuring a simultaneous execution of several platform-centric fusion applications, each having by definition the characteristics of the MSDF, Situation Threat Assessment (STA) and Resource Management (RM) of the corresponding platform. This architecture must provide the necessary resources to implement and test various information management rules and procedures (push/pull, prioritization and thresholding, etc.) inspired by the "Canadianization of Handbook V" (Combat System In-Service Support (CSIS) Task 109), which provides Canadian requirements for optimizing the implementation of a Maritime Tactical Picture (MTP) at sea, in compliance with AUSCANNZUKUS Handbook V "Guidelines for Maritime Information Management". The proposed architecture must also be consistent with the existing communication mechanisms (Link-11, Link-16, Link-22, Global Command and Control System (GCCS), etc.).

Figure 1 illustrates a basic network-centric data fusion system for two cooperating platforms (HALIFAX class frigate and the CP-140 Aurora surveillance aircraft).

This document presents the design and implementation of such a network-centric capability using the respective blackboard-based platform-centric MSDF/STA/RM capabilities.

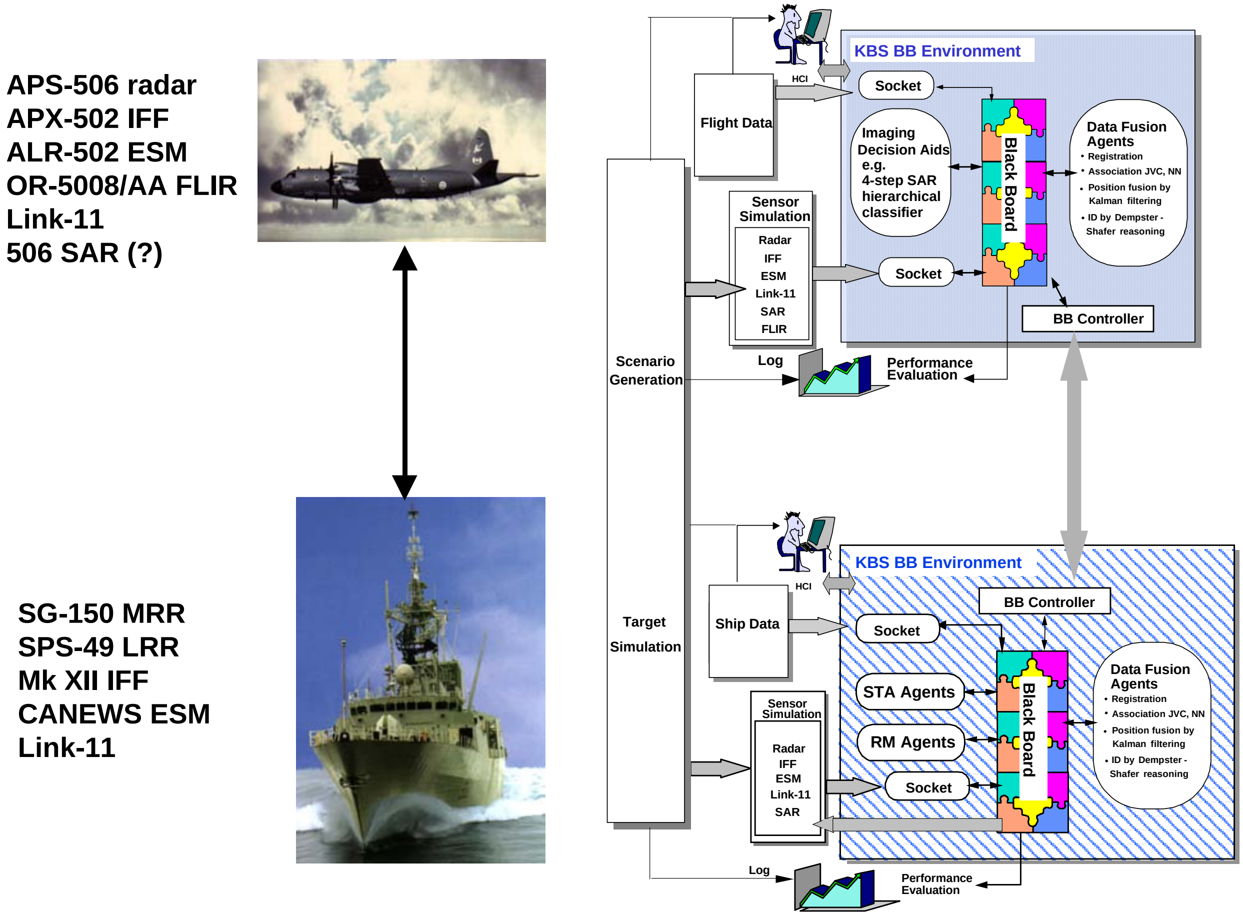



*Figure 1. Basic network-centric data fusion system involving two cooperating platforms using a knowledge-based system blackboard environment, such as Cortex*

This document is organized as follows:

a. Section 2 provides an overview of typical data fusion problems posed in decentralized data/information fusion networks and the algorithms that have been reported in the literature to overcome them. It also presents a high level description of the implementation of the network-centric data fusion architecture on LM Canada/DRDC Valcartier's Cortex.

b. Section 3 presents a high level overview of tactical datalink architecture and protocol and other considerations that impacted the test-bed infrastructure.

c. Section 4 details the requirements associated with the implementation of this architecture given the specificity of the communication channels (Link-11, Link-16, Link-22, GCCS)

d. Section 5 presents the detailed design of the implementation of a network-centric data fusion architecture on an existing knowledge-based system (KBS) jointly developed by LM Canada and DRDC Valcartier called Cortex

e. Section 6 presents the concepts of information prioritization for communications between participating units (PUs) for both datalink and point-to-point communications

f. Section 7 reports on the compatibility and availability of simulation environments as well as High Level Architecture (HLA) compliance

g. Section 8 reports on the performance of the chosen algorithms for track-to-track fusion, for classifiers for forward looking infrared (FLIR) imaging data from airborne platforms, and the fusion of such classifiers

h. Finally, Section 9 provides conclusions and recommendations

# 2. Concepts of decentralized fusion networks

Military operations are moving from platform-centric warfare to Network-Centric Warfare (NCW), whose organizing principle has its antecedent in the dynamic of growth and competition that have emerged in the modern economy. NCW derives its power from the strong networking of well-informed but geographically dispersed forces. The enabling elements are a high-performance information grid, access to all appropriate information sources, weapons reach and manoeuvring with precision and speed of response, value-adding command and control processes — to include high speed automated assignment of resources — and integrated sensor grids closely coupled in time to shooters and Command and Control (C2) processes. NCW is applicable to all levels of warfare and contributes to the coalescence of strategy, operations and tactics. It is transparent to mission, force size and composition, as well as geography.

These requirements motivate a close examination of decentralized data fusion architectures, i.e., a multi-sensor system in which there is no centralized communication, coordination or control. Instead, each sensor node is empowered with the capability to process data, communicate information and make local decisions. As a consequence, a decentralized data fusion system is a collection of processing nodes connected by communication links, where each node performs a specific computing task using information from nodes with which it is linked, but where no central node controls the network. None of these nodes has knowledge about the overall network topology. The most attractive properties of such a decentralized system are:

1. <u>Reliability</u>: the loss of a subset of nodes and/or links does not prevent the rest of the system from functioning. In a centralized system, however, the failure of a common communication manager or a centralized controller can result in immediate catastrophic failure of the system.

2. <u>Scalability</u>: nodes can be added or deleted by making only local changes to the network. The addition of a node simply involves the establishment of links from one or more nodes to the network. In a centralized system, however, the addition of a new node can change the topology in such a way as to require massive changes to the overall control and communication structure.

In a battlefield management application, one node might be associated with the acquisition of information from reconnaissance photographs, another with ground-based reports of troop movements, and another with the monitoring of communication transmissions. Information from the troop movements node could then be transmitted back to the reconnaissance photo node, which would use the estimated position of troops to aid in the interpretation of ambiguous features in satellite photos.

The most serious problem in decentralized fusion networks is the effect of redundant information. Specifically, pieces of information from multiple sources cannot be combined within most filtering problems unless they are independent or have a known degree of correlation (i.e., i.e., known cross-covariances). If communication paths are not strictly controlled, pieces of information may begin to propagate redundantly. When these pieces of information are reused (double-counted), the fused estimates produced at different nodes in the network become corrupted.

It is difficult to avoid this so-called data incest (or data-looping) problem without relaxing the constraints on the distributed data fusion architecture, which yields its compelling benefits. Utete (1998) has demonstrated that it is impossible to avoid this problem within the traditional framework of classic Kalman theory because of the statistical independence assumption.

Two approaches may then be considered to solve this problem:

1. Modify the decentralized data fusion architecture with respect to what is sent on the communication channel and where it is fused.

   This requires the implementation of some rules to control the flow of information. Even with this modification, cross-correlations between tracks seen by many platforms for the same apparent target must be taken into account when designing the filter used to combine the data (track-to-track fusion). A solution to this problem is the tracklets approach proposed by Drummond (Drummond, 2001). However, the tracklet interval imposes a delay in the updating process that may be detrimental to the decision-making process. Replacing track-to-track fusion with measurement-to-track fusion would solve the cross-correlation problem with a simultaneous update of the global and local track databases.

2. Solutions based on data tagging.

   To preserve the flexibility and scalability of the decentralized architecture, solutions based on data tagging cannot be considered. Currently, the two most popular data processing approaches are the distributed Bellman-Ford equations and the Covariance Intersection (CI) theory. For track fusion, CI provides an updated estimate even when the correlation between prediction and observation is unknown.

The requirements and design decisions taken for the implementation of the decentralized data fusion system within Cortex should provide the capability of studying both approaches. In decentralized data fusion networks, one has to decide how and where the processing is done.

This first part will focus on **how** the processing is done by providing in the following sections a detailed description of track-to-track fusion using tracklets and the CI approach. These descriptions will also define the nature of the information to be communicated, which may eventually influence **where** the computing tasks are distributed.

## 2.1 Track-to-track fusion

Drummond has proposed a classification of the algorithm architectures (*algorithm architecture*, meaning the sequence of processing functions or the processing chain) that are commonly used to perform data fusion from multiple sensors.

Table 1 below presents the four generic architectures proposed by Drummond.

*Table 1. Generic algorithm architectures for fusing data provided by multiple sensors*

| Type | Description |
|---|---|
| I | Single sensor processing |
| II | Track Fusion: also called hierarchical, federated, decentralized, distributed fusion or sensor-level tracking. Two variants exist: with or without feedback to the sensors. |
| III | Composite measurement fusion: sensors are synchronized. |
| IV | Central Fusion or Measurement Fusion, also called central-level tracking or centralized algorithm architecture |

### 2.1.1 Type I data fusion

This type of fusion corresponds to the processing performed by a single sensor (i.e., its own tracking capability). As there is no information sharing between sensors, this type of fusion will not be discussed.

### 2.1.2 Type II data fusion – track fusion

Track fusion (also called hierarchical, federated, decentralized or distributed data fusion), usually refers to a specific multi-sensor network architecture in which each sensor or platform provides local estimates (the output of each sensor's local data fusion module) to a data fusion centre that fuses them to produce global estimates.

Since sensor (platform) tracks are first formed locally and the track data are later used to update global (multiple sensor) tracks, each measurement is subjected to two association processes, first in the local measurement-to-sensor-track association forming the sensor (or platform) track, and second in the sensor-track-to-local-track association to update the local track database.

The first association process is performed within the processing units of the actual sensor, so we do not have control over the actual processing. However, the sensor tracks provided by the sensors of a given platform contribute to the formation of the **local track database** of the corresponding platform. They also contribute to the formation of the **global track database** of the platform which results from the fusion of the local track database with the tracks provided by other platforms.

The **local track database** is updated by the fusion of the sensor tracks of a given platform to the local tracks, as shown in Figure 2. In the current implementation within Cortex, the sensor tracks are not fused. A measurement from the sensor track is simulated by discarding the velocities from the state vector and building the covariance matrix associated with this synthetic measurement from the positional components of the state vector and nominal uncertainties of the corresponding sensor. This method avoids the cross-correlation that would result from the fusion of the sensor tracks provided by the multiple sensors (i.e., SPS-49 and SG-150) associated with the same target. As a consequence, the measurements synthesized from the sensor tracks are independent of each other and can be fused with the standard Kalman filter theory.

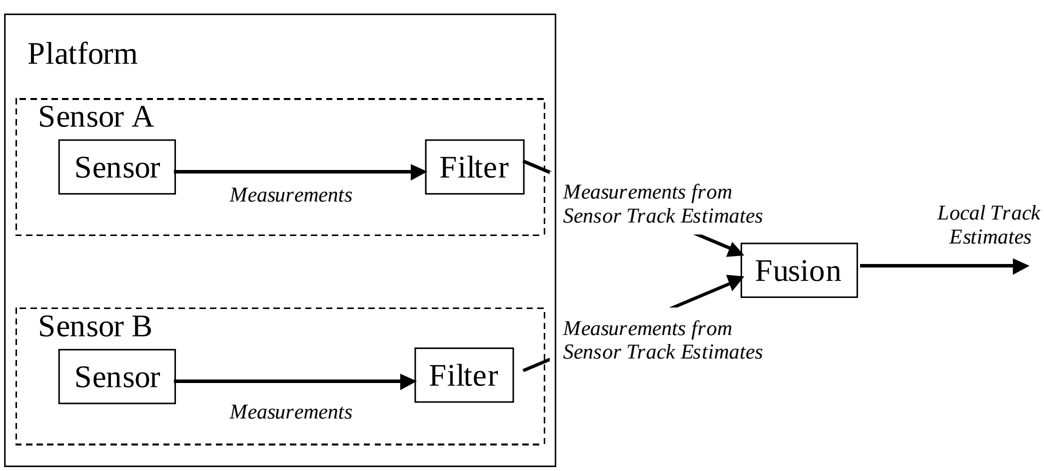


*Figure 2. Sensor-track fusion for the local track database*

The **global track database** is updated by the fusion of the local track database with tracks provided by other platforms as shown in Figure 3. In this case, the fusion of two tracks representing the same target is no simple matter, since the estimation error for each track might be cross-correlated. The solution proposed by Drummond is to fuse tracklets instead of tracks.

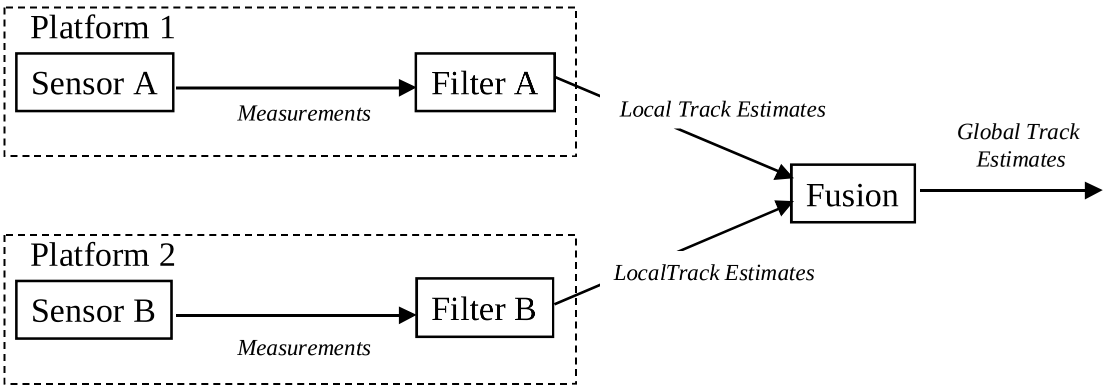



*Figure 3. Track fusion for the global track database*

**Tracks** or **tracklets** can be fused by rigorously using the same formalism. Note that two variants of decentralized data fusion have been proposed:

a. Type II fusion without feedback is the general formalism associated with decentralized track fusion.

b. Type II fusion with feedback provides better tracking performance, and studies have shown that the results are comparable to those obtained with centralized fusion. Some limitations inherent in the feedback itself will be pointed out below.

Each of these approaches is discussed in the following subsections.

### *Decentralized fusion without feedback*

For each local sensor or platform *i*, the measurement at time *t* is given by:

$$z_i(t) = H_i x(t) + v_i(t) \qquad i = 1, ..., N$$

Where $z_i(t)$ is the measurement vector of processor *i* at time *t*, $v_i(t)$ is a zero-mean white Gaussian noise with variance $R_i(t)$ and $H_i$ is the measurement matrix.

If the fusion agent collected all the sensor data, the global estimate would be

$$z\ (t) = H\ x(t) + v\ (t)$$

with

$$z(t) \equiv [z_1^T(t),...,z_N^T(t)]$$
$$H \equiv [H_1^T,...,H_N^T]$$
$$v(t) \equiv [v_1^T,...,v_N^T(t)]$$

and the covariance of the noise *v(t)* is given by:

$$R \equiv diag[R_1,...,R_N]$$

Estimation theory provides the local estimate for the sensor or platform *i* as:

$$\hat{x}_i(t\,|\,t) = \hat{x}_i(t\,|\,t-1) + P_i(t\,|\,t)H_i^T R_i^{-1}[z_i(t) - H_i\,\hat{x}_i(t\,|\,t-1)]$$

with covariance given by:

$$P_i^{-1}(t\,|\,t) = P_i^{-1}(t\,|\,t-1) + H_i^T R_i^{-1} H_i$$

The global estimates with all sensor data is given by:

$$\hat{x}(t\,|\,t) = \hat{x}(t\,|\,t-1) + P(t\,|\,t)H^T R^{-1}[z(t) - H\,\hat{x}(t\,|\,t-1)]$$

$$P^{-1}(t\,|\,t) = P^{-1}(t\,|\,t-1) + H^T R^{-1} H$$

The fusion of local estimates and covariance should lead to the global estimate and covariance.

The expression of the local and global covariance leads to:

$$P^{-1}(t\,|\,t) = P^{-1}(t\,|\,t-1) + \sum_{i=1}^{N}[P_i^{-1}(t\,|\,t) - P_i^{-1}(t\,|\,t-1)] \tag{1}$$

Similarly, the global estimate can be obtained with

$$P^{-1}(t\,|\,t)\,\hat{x}(t\,|\,t) = P^{-1}(t\,|\,t-1)\,\hat{x}(t\,|\,t-1) + \sum_{i=1}^{N}[P_i^{-1}(t\,|\,t)\,\hat{x}_i(t\,|\,t) - P_i^{-1}(t\,|\,t-1)\,\hat{x}_i(t\,|\,t-1)] \tag{2}$$

These two equations are the basic equations when communication is one-way, i.e., there is no feedback from the global estimator to the local estimators.

At each *t*, the sensor (platform) *i* transmits the state estimate $\hat{x}_i(t\,|\,t)$ and prediction $\hat{x}_i(t\,|\,t-1)$, as well as the covariance $P_i(t\,|\,t)$ and prediction $P_i(t\,|\,t-1)$ to the data fusion centre, which computes $\hat{x}(t\,|\,t)$ and $P(t\,|\,t)$ from Equations (1) and (2). The terms $P_i^{-1}(t\,|\,t-1)$ and $P_i^{-1}(t\,|\,t-1)\,\hat{x}_i(t\,|\,t-1)$ are already

present in $P^{-1}(t|t-1)$ and $P^{-1}(t|t-1)\hat{x}(t|t-1)$, respectively, and thus should be removed from $P_i^{-1}(t|t)$ and $P_i^{-1}(t|t)\hat{x}_i(t|t)$ in Equations (1) and (2) to avoid double-counting.

This formalism permits the fusion of only the local track information related to the new track update in the global estimates. It prevents the introduction of cross-correlation between the global track estimates and local track estimates.

### *Decentralized fusion with feedback*

Figure 4 shows a simple decentralized tracker as an example of decentralized data fusion architecture with feedback. Each sensor provides measurements to its own tracking filter, whose output is combined in the fusion filter to produce a fused track. It has been demonstrated that the feedback significantly increases tracking quality, and comes close to that obtained with a centralized data fusion system (type IV data fusion in Table 1).

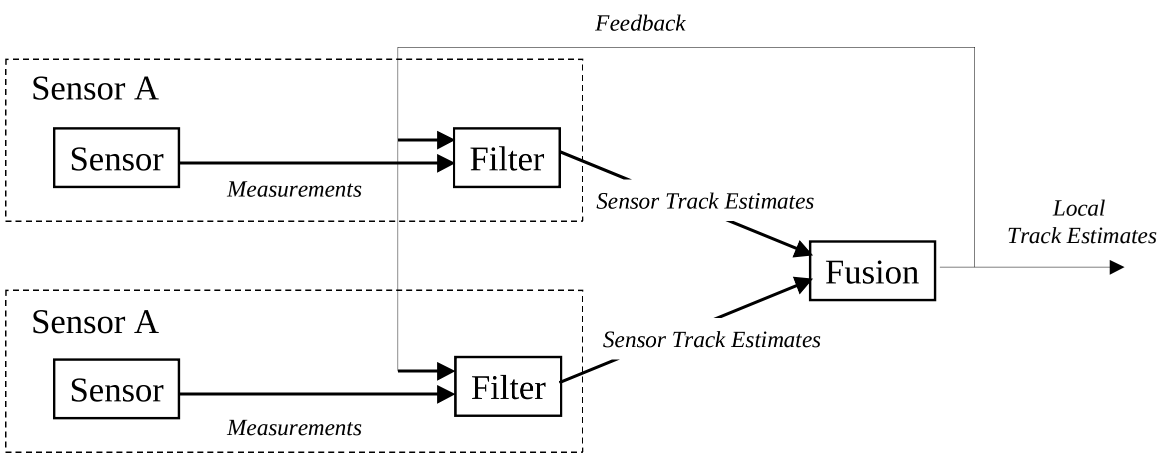



*Figure 4. Decentralized fusion with feedback*

Some of the estimates and covariances may not have to be communicated to the data fusion centre. For example, because $\hat{x}_i(t|t-1)$ is the extrapolation of $\hat{x}_i(t-1|t-1)$ at time *t*, it can be computed at the data fusion centre from the dynamic equation:

$$x(t+1) = F(t)x(t) + G(t)w(t) \qquad t = 0,1,2,...$$

where *x(t)* is the state vector of the target at time t, *w(t)* is a white Gaussian vector noise with variance matrix *Q(t)*,with *F(t)* and *G(t)* being known.

Similarly the covariance $P_i(t|t-1)$ can be extrapolated from $P_i(t-1|t-1)$ at the data fusion centre.

The matrices $P_i(t|t)$ need not be communicated at all when they do not depend on measurements, for example, when all the dynamic and observation equations are known *a priori* (see restrictions from Drummond).

When there is feedback, the fusion centre broadcasts its latest estimate to the sensors (platforms):

$$P_i(t-1|t-1) = P(t-1|t-1)$$

$$\hat{x}_i(t-1|t-1) = \hat{x}(t-1|t-1)$$

The fusion equations with feedback are obtained by substituting the latest global estimate in Equations (1) and (2), yielding:

$$P^{-1}(t|t)\hat{x}(t|t) = \sum_{i=1}^{N}[P_i^{-1}(t|t)\hat{x}_i(t|t) - (N-1)P^{-1}(t|t-1)\hat{x}(t|t-1)] \tag{3}$$

$$P^{-1}(t|t) = \sum_{i=1}^{N}[P_i^{-1}(t|t) - (N-1)P^{-1}(t|t-1)] \tag{4}$$

However, by propagating the global estimates into sensor filters, the feedback introduces a cross-correlation that must be dealt with to produce consistent measurements. Several approaches may be considered to circumvent this problem:

1. Tracklets approach: which divides old and new information, using only the new information gained since the last update in the fusion process.

2. Selective Position Fusion with covariance matrix approach: which updates the positional track state and the covariance matrix with the latest report from the network (note that the complete covariance matrix should be available from the remote sources so that fusion is not performed and no cross-correlation is introduced).

3. Selective Position Fusion with Track Quality approach: which quantizes the covariance matrix reported by Link-11 via a number between 1 and 15 and maps it into a circular area of uncertainty and a covariance matrix.

4. Covariance Intersection approach: which is a fusion method producing consistent estimates for any cross-correlated data without requiring any knowledge about the cross-correlation.

In conclusion, it should be remembered that using feedback might also be risky. If one sensor malfunctions and produces erroneous estimates, they will be spread through the output of the other sensors' filters. It is likely that the global track estimates will be corrupted. By propagating the erroneous global track estimates into the sensor filters, the global track estimates will become unreliable.

### 2.1.3 Type III data fusion

This type of fusion is also called Composite Measurement Fusion. The sensors (platforms) are synchronized in such a way that measurements from different sensors (platforms) for a target are taken at the same time. The measurements from the various sensors (platforms) that appear to be from the same target are first combined to form a composite measurement, which is then used to update the global tracks at the data fusion centre.

Each measurement is again subjected to two successive association processes: first in measurement-to-measurement from different sensors (platforms) to form the composite measurement, and second in composite-measurement-to-global-tracks to update the global tracks. A special case of this approach is fuse-before-detect.

### 2.1.4 Type IV data fusion

This type of fusion is also called Central Fusion or Measurement Fusion, Central-Level Tracking or Centralized Algorithm Architecture. Measurements from various sensors are used to update the global tracks. A frame of measurement from one sensor is processed with the latest global tracks, then the filter updates the global tracks. Once the processing of this frame is completed, a frame of measurement from another (or the same) sensor is processed with the updated global tracks and so on. This process is continued as new frames of data become available to the system as a whole. Each measurement is subjected to only one association function, a measurement-to-track association employing global tracks based on the prior data from all sensors.

## 2.2 Updating the global track database – track fusion

The following sections describe the two approaches considered for track fusion:

1. The Tracklets approach proposed by Drummond removes the cross-correlation between the tracks by fusing only the information acquired since the last update.

2. The Covariance Intersection approach is a new data fusion method that produces consistent estimates for any cross-correlation without actually knowing the cross-correlation.

### 2.2.1 Computing tracklets from the local tracks

Many variants of the tracklets approach exist, some being fairly equivalent. Two different methods for computing a tracklet were implemented and tested in the test-bed:

1. The inverse Kalman Filter

2. The inverse Information Filter

In order to be as general as possible, each fusion node broadcasts only tracks from its track database. No tracklet is broadcast. The receiving fusion node then computes tracklets for each track received from each node. Tracklet computation is performed at the Global Multi-Sensor Data Fusion (MSDF) level for each fusion node. To perform the various tracklet computations described above, Global MSDF needs to store the latest reported track state and covariance matrix, and the last computed tracklet, for each track reported by each fusion node.

Tracklet methods are applied only when the received data contains cross-correlation. When data is received from its Local MSDF, the global fusion node replaces the related global track state with the received track state. In fact, since fusion was already performed by Local MSDF, there is no need to compute an equivalent measurement and fuse it again, as repeating these computations would be time-consuming. At the beginning of the fusion process, the tracklet computation is performed using the new buffer of reported tracks. The tracklet is computed only if all information needed is available for the given reported track, i.e.,

- the latest reported track (for the inverse Information Filter), or

- the last tracklet (for the inverse Kalman filter).

If this information is not available, the system uses the latest report as a tracklet to initialize the tracklets processing. In the Reported Track Data Store, the computed tracklet is stored with the new reported track, replacing previously stored information. This information will be available for the next processing.

Prior to positional fusion, the gating process is performed on reported tracks and global tracks. The computed tracklets are not used during gating. Track-to-track association is then performed to determine which reported track is associated with which global track. Then, each confirmed track / track pair is fused using one of the position fusion algorithms, e.g., the Extended Adaptive Kalman Filter (EAKF) or the Interacting Multiple Model (IMM). The inputs to these fusion algorithms are the computed tracklets and the time-updated global tracks.

The following sections describe how the tracklets are computed in the two different approaches.

### *Inverse Kalman filter*

The inverse Kalman filter method consists of computing an equivalent measurement (tracklet) for each track update received from a particular source. The global tracker does not receive the original radar contact data, but does receive the results of the tracking performed by the reporting unit. The new state vector of the reported track received at time $t_n$ is denoted by $X_n^j$, while the covariance matrix at the same time is denoted by $P_n^j$. The superscript *j* denotes the reporting source of the track. In order to compute the new tracklet at time $t_n > t_m$, the last computed tracklet state vector $u_m^j$ and covariance matrix $U_m^j$ from source *j* at time $t_m$ must be time-updated to $t_n$ using the following equations

$$\Delta t = t_n - t_m$$

$$u_{n|m}^j = \Phi(\Delta t) u_m^j$$

$$U_{n|m}^j = \Phi(\Delta t) U_m^j \Phi^T(\Delta t)$$

where $\Phi(\Delta t)$ is the state transition matrix and the time-updated tracklet is noted as ($u_{n|m}^j, U_{n|m}^j$). Note that the process noise is assumed null for the time update of the covariance matrix of the tracklet. The following equations compute the new tracklet state vector $u_n^j$ and the new covariance matrix $U_n^j$ for the source $j$ at time $t_n$.

$$u_n^j = u_{n|m}^j + U_{n|m}^j \left[ U_{n|m}^j - P_n^j \right]^{-1} \left[ X_n^j - u_{n|m}^j \right]$$

$$U_n^j = U_{n|m}^j \left[ U_{n|m}^j - P_n^j \right]^{-1} U_{n|m}^j - U_{n|m}^j$$

Note also that a tracklet has the same dimension as a track, i.e., it contains both the position and the velocity components.

### *Inverse Information Filter*

The inverse Information Filter uses the same approach as the inverse Kalman filter when computing an equivalent measurement. But unlike the inverse Kalman Filter, the inverse Information Filter is more general and takes into account process noise. As with the inverse Kalman filter, the newly received information is the state vector $X_n^j$ and its covariance matrix $P_n^j$ from source *j* at time $t_n$. The tracklet computation needs the previously received information ( $X_m^j, P_m^j$ ) at time $t_m$ to be propagated to time $t_n$. The time-updated information state vector and its covariance matrix are noted $X_{n|m}^j$ and $P_{n|m}^j$, respectively. The following equations compute the tracklet's state vector $u_n^j$ and covariance matrix $U_n^j$ for the source $j$ at time $t_n$.

$$U_n^j = \left[ P_n^{j^{-1}} - P_{n|m}^{j^{-1}} \right]^{-1}$$

$$u_n^j = U_n^j \left[ P_n^{j^{-1}} X_n^j - P_{n|m}^{j^{-1}} X_{n|m}^j \right]$$

where the time update is given by

$$\Delta t = t_n - t_m$$

$$X_{n|m}^j = \Phi(\Delta t) X_m^j$$

$$P_{n|m}^j = \Phi(\Delta t) P_m^j \Phi^T(\Delta t) + Q$$

where $\Phi(\Delta t)$ is the state transition matrix and *Q* is the process noise estimation for the reported track. *Q* is variable and based on the speed variance of the reported track. The speed variance is computed from a linear regression of the last "*N*" speed estimates of the reported track.

## 2.2.2 Covariance Intersection (CI)

The Covariance Intersection method is a fusion method that produces consistent estimates for any cross-correlation without actually knowing the cross-correlation.

Let's assume that the local state estimates $\hat{x}_A$ and $\hat{x}_B$ are provided by two platforms A and B tracking the same target. Let's also assume that the true values are $\bar{x}_A$ and $\bar{x}_B$ (relative to platforms A and B, respectively). The deviations from the true values are $\tilde{x}_A = \hat{x}_A - \bar{x}_A$ and $\tilde{x}_B = \hat{x}_B - \bar{x}_B$. The covariance matrices are defined as:

$$\bar{P}_{x_A x_A} = E\{ \tilde{x}_A \tilde{x}_A^T \} \qquad \bar{P}_{x_A x_B} = E\{ \tilde{x}_A \tilde{x}_B^T \} = E\{ \tilde{x}_B \tilde{x}_A^T \} = \bar{P}_{x_B x_A} \qquad \bar{P}_{x_B x_B} = E\{ \tilde{x}_B \tilde{x}_B^T \}$$

These values are not known but are approximated by $P_{x_A x_A}$ and $P_{x_B x_B}$, where $P_{x_A x_B}$ is approximated to be zero. The Kalman filter uses a linear fusion of the state estimates which can be written as:

$$\hat{x} = w_A \hat{x}_A + w_B \hat{x}_B$$

$$P = w_A P_{x_A x_A} w_A^{\,T} + w_A P_{x_A x_B} w_B^{\,T} + w_B P_{x_B x_A} w_A^{\,T} + w_B P_{BB} w_B^{\,T}$$

$w_A$ and $w_B$ are weights chosen to minimize the trace of $P$.

For two-dimensional local state estimates $\hat{x}_A$ and $\hat{x}_B$**,** the fusion update can be schematized by plotting their covariance ellipses. Regardless of the value of $P_{x_A x_B}$, $P$ will always lie within the intersection of $P_{x_A x_A}$ and $P_{x_B x_B}$. An algorithm that finds a $P$ that encloses the intersection of the two ellipses will always be consistent whatever the value of $P_{x_A x_B}$. This is the principle of the Covariance Intersection (CI), also called Gauss Intersection.

Figure 5 illustrates the CI principle. The two solid-line ellipses are the covariances $P_{x_A x_A}$ and $P_{x_B x_B}$. The broken-line ellipse represents the covariance $P$ computed using CI.

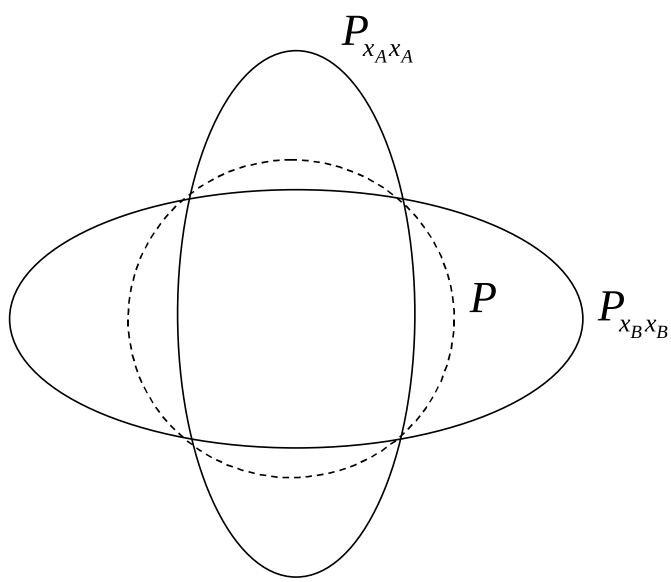



*Figure 5. The Covariance Intersection principle*

The CI method takes a convex combination of the inverse of means and covariances:

$$P = [wP^{-1}{}_{x_A x_A} + (1-w)P^{-1}{}_{x_B x_B}]^{-1}$$

$$\hat{x} = P\,[wP^{-1}_{x_A x_A}\,\hat{x}_A + (1-w)P^{-1}_{x_B x_B}\,\hat{x}_B]$$

$w$ is chosen to minimize $P$ so that it will enclose the intersection as tightly as possible.

It can be proven that the CI provides a consistent $P$ if $\hat{x}_A$ and $\hat{x}_B$ are consistent for any choice of $P_{x_A x_B}$ or $w$.

These equations can easily be generalized to the fusion of more than two local track state estimates as:

$$P_{sum} = \sum_n w_n P_n^{-1}; \qquad \sum_n w_n = 1$$

$$\hat{x} = P_{sum} \sum_n w_n P_n^{-1}\,\hat{x}_n$$

The efficiency of the CI method depends on the choice of $w$. To gain the most information out of the original estimates, $w$ should be chosen in such a way that the resulting covariance is as small as possible while enclosing the intersection. To minimize $P$, it is necessary to define a norm of the size of $P$. There are several norms that may be used, e.g., the determinant, the trace, the Frobenius norm (defined as $\sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}\left|P_{i,j}\right|^2}$), or the maximum value or eigenvalue of $P$. Choosing a norm is a compromise between accuracy and computing burden. A norm considering all matrix elements (like the determinant or the Frobenius norm) is likely to produce better estimates than the norms based on the trace or maximum values.

Using the determinant as the norm, one must find the value of $w$ that satisfies:

$$\min_{w\in[0,1]} \frac{1}{\det(wP^{-1}_{x_A x_A} + (1-w)P^{-1}_{x_B x_B})}$$

The implementation of this minimization process will be easier if an analytical formulation of det( $P$ ) can be found.

While CI produces consistent estimates for any $w$, the optimization of $w$ will optimize the performance of the fusion process.

## 2.3 Updating the global track database – identity fusion

Sensor measurements may also contain target attributes related to the type, allegiance and offensiveness of the target. This information can be used to solve ambiguous measurement-to-tracks associations. There is no theory to decide on the right way to use the identity for association; different approaches exist based on *ad hoc* choices.

The approach currently used consists of penalizing the positional probability of association $P_a$ with a measure of the conflict between the reported attribute of highest mass noted

$\sup(mass[ID\_reports])$ and the most probable target identity (identity proposition of highest mass in the Dempster-Shafer sense noted $\sup(mass[ID\_track])$ ). If the intersection of the associated propositions is empty, the propositions are conflicting and the association probability based on attributes is defined as:

$$p\_attr = 1 - \{\sup(mass[ID\_reports] \times \sup(mass[ID\_tracks])\}$$

If the intersection of the associated propositions is not empty, the propositions are not conflicting and the association probability is set to unity.

The mathematical expression of the total association probability that is used to populate the assignment matrix is:

$$p = 100 \times [1 - p\_pos \times p\_attr \times p\_prom]$$

where $p\_prom$ stands for the promotion probability which is set by default to predetermined values (1, 0.95, and 0.9 respectively for a firm, tentative, and initiated track). The normalization factor 100 is set only for convenience. The resulting probability represents the probability of non-association.

This fusion method works reasonably well for reports that are not delayed in time. In the next section a totally different approach will be detailed for the fusion of links and GCCS reports that can be made available for fusion with a delay that ranges from a couple of minutes to several hours.

## 2.4 Multi-blackboard design for track-level fusion

The data fusion architecture necessary to study the exchange of information between cooperative platforms has been designed and implemented within a KBS blackboard (BB) environment, called Cortex, originally developed at LM Canada, jointly with DRDC Valcartier. The objectives of any NCW system (or simulated system) are to ensure a timely exchange of secured (uncorrupted) information through the information grid. The decentralized data fusion system proposed in this project is consistent with the following NCW requirements:

1. Network topology is unknown, i.e., the cooperating platforms do not need to know how many they are, who they are and where they are

2. The circulation of information will not be affected if a platform is not capable of communicating or is destroyed

To achieve this goal, each platform will maintain two track databases: the local track database and the global track database.

1. The local track database contains the local track estimates produced by the fusion of the measurements created from the local sensor tracks. The single platform MSDF test-bed stores tracks in a local track database. This local track database contains tracks of various types (XY, XYZ, BO) using various filters (Interacting Multiple Models (IMMs), variations of the Kalman filter), and various data association methods (Jonker-Volgenant-Castanon (JVC), Nearest Neighbour (NN), Joint Probabilistic Data Association (JPDA), Multi-Hypotheses). A local picture is produced by the display of the local track database that is refreshed on a track-by-track basis once one track has been updated.

2. The global track database of each platform will be updated by information provided by each platform's local track database and by information provided by the local track database of other platforms. The information shared by platforms will emulate a Link-11/16/22 broadcast.

Figure 6 is an illustration of a decentralized data fusion network involving two platforms. Each platform locally computes a local track estimate that is shared with others and fused to their global track estimates.

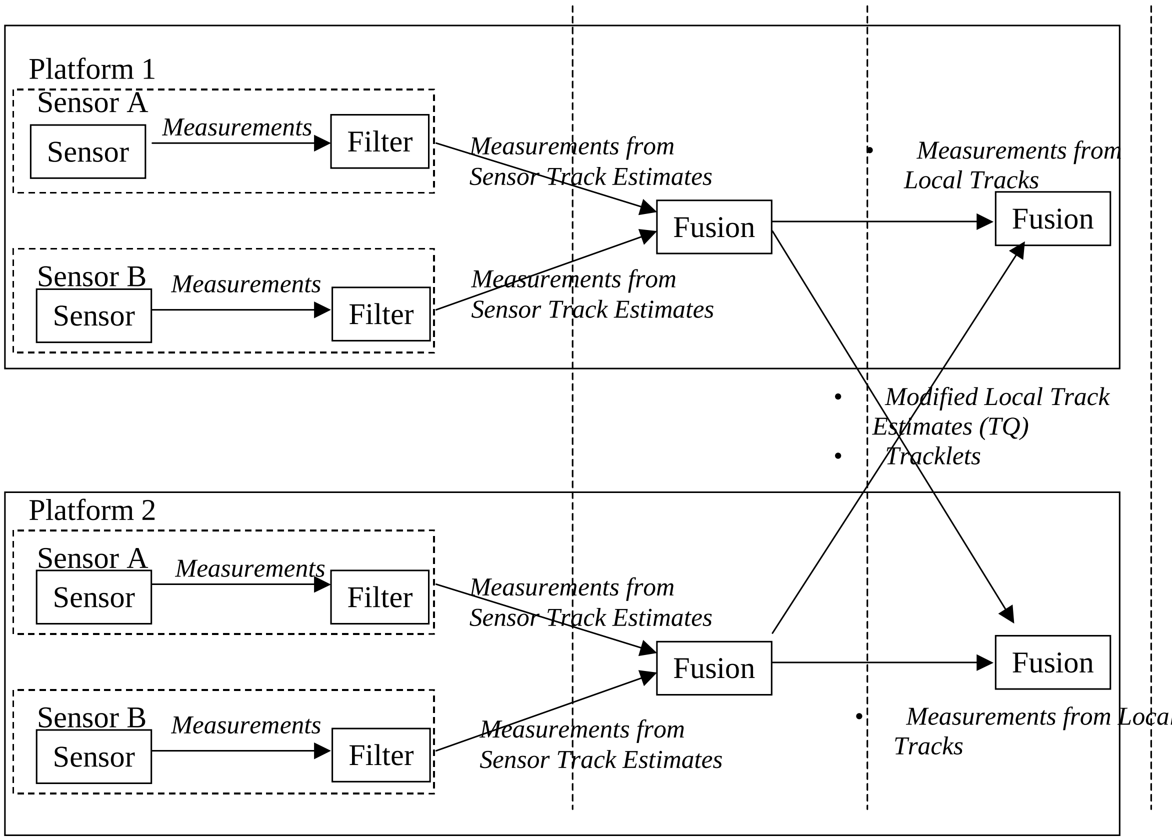



*Figure 6. Decentralized data fusion on Cortex – high-level diagram*

### 2.4.1 Data fusion at the local track database level

As mentioned in Section 2.1.2, measurements are created from sensor tracks, i.e., the data fusion at the local track database level is a measurement-to-track fusion. The covariance matrix associated with the measurement is built from the positional part of the sensor track and the nominal uncertainties of the sensors. Measurements can be considered independent, and no correlation exists between them and the local tracks.

The standard Kalman filtering theory can then be used to fuse positional target estimates at the local track database level. Measurements may be late with respect to the last track update due to the buffering of asynchronous sensors. This can be handled by a noiseless retrodiction.

Target attributes provided by local sensors can be introduced in the data association process by using the penalization of the positional probability of association described in Section 2.3.

The fusion of target attributes with the identity of local tracks is performed using Dempster-Shafer evidence theory, in a truncated form in order to prevent algorithmic explosion.

### 2.4.2 Fusion of the local and global picture of a given platform

Each platform will have to update its global track database with the track data stored in its local track database.

At the creation of the global track database a simple copy of the local track database is made. The time at which the data has been transferred is saved.

Then, after a given time interval, the global track database must be refreshed by the information collected in the local track database. This is a track fusion problem. Global tracks may also be refreshed by information collected by other platforms (see Section 2.4.3). Still, the track data fusion to be employed to update the global track database with the local track database must address the correlation (data incest) issue to avoid fusing the same information more than once.

There are two ways of solving this problem: either choose a track-to-track fusion using the tracklet approach, or update the tracks of the global track database with the sensor track measurement simultaneously with the update of the local track database. The implementation of these two approaches is detailed in the following paragraphs.

#### *The tracklet approach*

The time duration between successive updates of the global track database defines the tracklet interval. Two cases must be considered:

1. Track creation, which applies to tracks created in the local track database after the last update of the global track database, is performed by simply copying the local track in the global track database
2. Track maintenance, which applies to all tracks that existed at the time of the last update of the global track database, requires dealing with the correlation between the local track state estimates at the current time of update and the last time of update and the global track state estimate. This may be done by computing the tracklets associated with each local track (see Section 2.2.1) and fusing them to their corresponding equivalent in the global track database using the classical Kalman filter formalism.

Some tuning of the tracklet interval will be necessary to ensure the global track estimates are sufficiently accurate.

Figure 7 represents the fusion of the local and global track databases for two tracks. At the beginning when no correlation needs to be considered, the local track estimates $\hat{x}_t^{Track\ number}$ are sent ($t$ is the time). For later updates, the tracklets $y_t^{Track\ number}$ are computed for each track and fused.

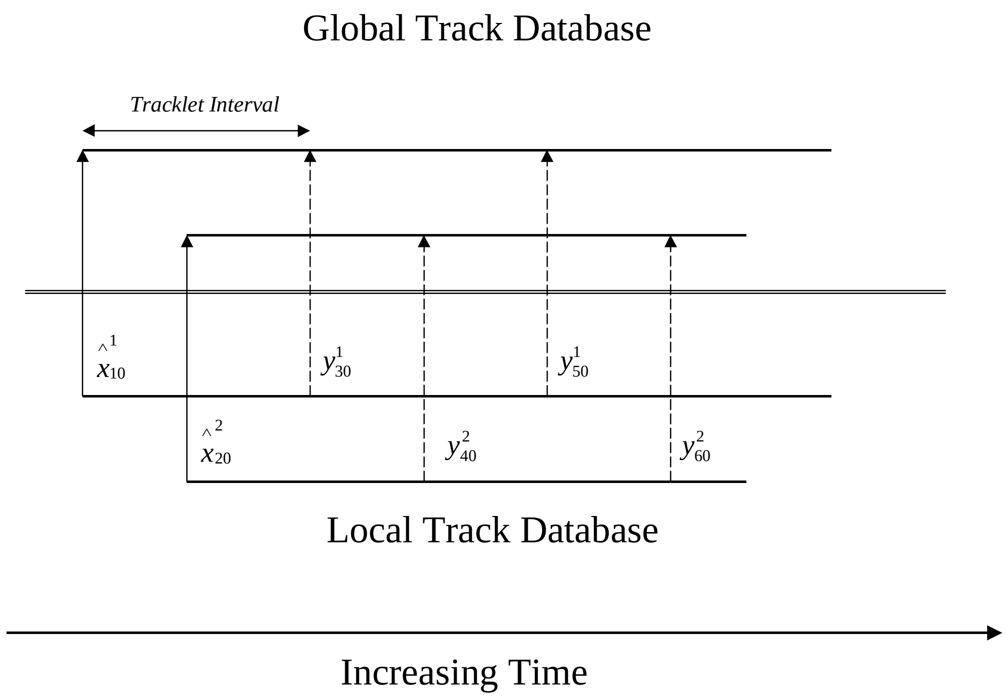



*Figure 7. Fusion of local and global track databases for a given platform*

#### *Fusion of sensor track measurement to globaltTracks*

It is desirable to avoid delays between the update of the local track database and the global track database. Once information has been fused locally it should be made available globally as soon as possible.

Once a sensor track measurement has been selected to update a given local track (gating), the update of the local track and its corresponding global track with the sensor track measurement can be performed simultaneously. This approach would avoid delays between the local and global track updates and totally eliminate the correlation inherent in track-to-track fusion.

### 2.4.3 Fusion of the local picture of a given platform to the global picture of another platform

To simultaneously update its global track database by its own local track database, every platform would have to fuse the track state estimates provided by the other platforms. This is usually done using Link-11/16/22 broadcasts. The decentralized data fusion system proposed in this project would broadcast the information with the implementation of a NetManager responsible for the polling of each platform in turn.

When polled for the first time, a platform sends its local track state estimates to other platforms for fusion with their global track database:

1. If the track already exists in the global track database of many platforms, no correlation exists with the local track estimates provided by a given platform that reports for the first time. The fusion of the newly distributed track state estimates to an existing track can be performed directly using Kalman filtering.

2. If the track does not exist in the global track database, it is created using the newly distributed track estimates.

When polled for the second time and later, a platform must send:

1. The local track state estimates for tracks that have been created since the last update.

2. The tracklets computed for each track previously distributed if tracklets is the selected data fusion approach. The elapsed time between successive polls of the same platform will determine the tracklet interval.

3. The track state and covariance if CI is the selected data fusion approach.

4. The track measurements if measurement-to-track is the selected data fusion approach.

Note that the Link-11/16/22 format using a reduction (discretization) of the track covariance to the track quality is more compatible with the two latter approaches (CI or measurement-to-track). Moreover, Link-11/16/22 reports are delayed in time, so its use for a positional update of a target, which is already tracked locally, is questionable. However, using it for the identity update of a target tracked locally is more acceptable.

### *How to deal with "late" reports from remote platforms*

As discussed earlier, reports from remote platforms about targets that are tracked locally are likely to be late, i.e., they will carry information that has been measured at a past time when compared to the current time of the local estimates. Link-11/16/22 reports may be minutes late, and for GCCS data, hours.

As a rule of thumb we propose to solve the assignment problem by following the approach described in Section 2.3 and making a track prediction from the closest track update in the history just before the time of the remote report.

If the delay between the incoming remote report and the time of the last update of the global track to be updated with the report is:

1. Smaller than the time taken by the slowest sensor of the platform that receives the reports to complete three (default value) scans:

    a. Make a positional update of the global track using either the tracklet, CI or measurement-to-track approach

    b. Perform the attribute fusion (if applicable) using Dempster-Shafer rules

2. Greater than the time taken by the slowest sensor of the platform that receives the reports to complete three (default value) scans:

    a. Do not bother with a positional update that would be insignificant and largely inaccurate

    b. But perform the attribute fusion (if applicable) using Dempster-Shafer rules as usual.

### *Link-11/16/22 and GCCS contribution to the global track database*

The data fusion at the global track database level for a given platform will consist of

1. The track fusion of the global track from the local track using either the tracklet approach or the fusion of sensor-track-measurement approach

2. The track fusion of global track and Link-11/16/22 remote data using either the tracklet, CI or measurement-to-track approach if the reported information is not too old

3. The track fusion of global track with GCCS data that is likely to be reduced to an attribute fusion if the reported data is related to an already tracked target.

As a consequence, it is expected that the data fusion of positional data at the global track database will consist mainly of the track fusion with the local track database. The contribution of the positional fusion from Link-11/16/22 reports will be less significant, and from GCCS probably irrelevant. The association of Link-11/16/22 or GCCS reports to tracks of the global track database is likely to be more useful for attribute fusion than for positional fusion.

### 2.4.4 Dead reckoning

Before sending its local track database, the polled platform may perform a time propagation of all the tracks to synchronize them to a unique time. Since not all tracks are propagated periodically but only when they are updated, tracks would be propagated to the time of broadcast. The impact of such a capability must be investigated further.

### 2.4.5 Global track selection

Once a platform distributes a set of tracks, the other platforms have to select the global tracks that may be associated with them. It is suggested that each platform receiving the track data compute the volume within which the distributed tracks lie. To do that, each track of the global database is first synchronized in time (dead reckoning) with that of the reported tracks. This step would be easier if all the reported tracks had been previously synchronized to the broadcast time as suggested in the previous section. Then, a bearing interval and a range interval that are relative to the receiving platform may be used to define this volume, making it easier for the receiving platform to select the global tracks that may gate with the distributed tracks.

### 2.4.6 Track-to-track gating

This is required for both tracklets and CI data fusion approaches. Once a set of global tracks has been selected from the global track database as candidates for fusion, the state difference $\tilde{y}_{T_iT_j}$ between tracks *i* and *j* is computed as

$$f(\tilde{y}_{T_iT_j}) = e^{-d_{T_iT_j}/2} / \left[ (2\pi)^{M/2} \sqrt{\left| P_{T_iT_j} \right|} \right]$$

where

$$d_{T_iT_j} = \tilde{y}_{T_iT_j}^T . P_{T_iT_j}^{-1} . \tilde{y}_{T_iT_j}$$

$$\tilde{y}_{T_iT_j} = \hat{x}_{T_i} - (y_{T_j})_n^k$$

$$P_{T_iT_j} = P_{T_i} + (Y_{T_j})_n^k$$

The global state estimates for track *i* is noted $\hat{x}_{T_i}$ , and the tracklet for track *j* is computed by platform *k* at time *n* is $(y_{T_j})_n^k$. The dimension of the state vector is noted *M*.

The positional probability of association can then be computed, and then penalized by the conflicts between attributes as described in Section 2.3. The assignment matrix is built in a similar way to the classical contact-to-track association and solved by the NN or JVC algorithms.

### 2.4.7 Local vs. global track numbering

Appropriate control of local and global track numbering may slightly decrease the computer load in avoiding unnecessary gating processes.

## 2.5 Conclusion

This section presented an overview of the various data fusion problems encountered in decentralized architecture and the methods that have been proposed in the literature to solve them.

CI is a very attractive decentralized data fusion approach because of its simplicity and the fact that it permits the use of all the potential of a fully decentralized data fusion network without the bother of potential correlation between fused data and measurements.

The tracklets approach is simpler and somewhat safer than CI.

However, both approaches may be revealed as overkill when compared with a more classical measurement-to-track data fusion approach, specifically when measurement delays and measurement accuracy are taken into consideration. The network-centric data fusion architecture to be built must be capable of handling all these possibilities. Specific scenarios will have to be defined to challenge these approaches, and specific Measures of Performance (MOPs) will need to be defined in order to assess performance.

# 3. High-level concepts of the test-bed infrastructure

The test-bed must establish and demonstrate a baseline system that will help in the study of the communication process between platforms and the impact it has on joint situational awareness. The three major aspects of the communication process between platforms are:

1. When intelligence should be communicated

2. What should be the content of the communication

3. How—which format, and with whom—information should be shared

The test-bed will not simulate or emulate any particular tactical datalink system. The test-bed will have to offer enough flexibility to study more than one communication protocol, and not limit data exchange based on existing tactical datalinks, data types, field size, degree of precision, etc.

The following sections address some elements of network communication that need to be accounted for when defining the implementation requirements for the test-bed.

## 3.1 Tactical datalinks

The purpose of a tactical datalink is to support real-time surveillance and C2 information exchange among various C2 platforms and weapon platforms to enhance mission capabilities and performance. This ensures that a consistent tactical picture is available to all units in the Link-11/16/22 network.

The following subsections provide a high-level overview of the architecture and protocol differences between currently available/planned tactical datalinks:

1. Link-11, which is currently used by the Canadian Navy on the HALIFAX class ships and which will therefore receive special attention

2. Link-16, which is the Tactical Data Link of choice of the US Department of Defense, is more succinctly described

3. Link-22, which is the next-generation NATO Tactical Data Link, also referred to as the NATO Improved Link Eleven (NILE), will also be described extensively, in case of NATO coalition missions.

### 3.1.1 Link-11

Link-11 uses a polling protocol and netted architecture. A net is an ordered conference whose participants have common information needs or similar functions to perform. A net operates under the supervision of a controller, who permits access and maintains circuit discipline.

The Link-11 net is normally operated according to the Roll Call protocol. Participating units (PUs) transmit all data eligible for reporting when the Net Control Station (NCS) polls them. After transmission, they revert to the receiving mode while, one by one, the other PUs transmit their data. This cycle continues until all PUs have had the opportunity to transmit data, and then it is repeated. The time required to poll all PUs and transmit all their eligible data at least once is known as the net cycle time. At any given time, a PU is either transmitting or receiving data on a single Link-11 net.

### *Track Quality (TQ) assignment*

TQ is used to determine which system has the best data.

In Link-11 TQ is a numerical value from 0 to 7. The value 0 indicates a non-real-time report. Values 1 to 7 indicate different degrees of reliability of the positional data, where 7 represents the highest track quality. A specific positional accuracy range (in square miles) defines each TQ value. TQ values are computed for air tracks in accordance with the definitions of the TQ fields for the link M2 message. TQ values are computed for surface tracks in accordance with the definitions of the TQ fields for the link M3 message.

A track report is identified as a non-real-time report if any of the following conditions is true:

1. The track data originate from a non-Tactical Data System (non-TDS)
2. The track data have been relayed by other than a datalink interface
3. The track data have been derived from other than integrated active sensors.

### *Track Reporting Responsibility ($R^2$)*

For air and surface tracks, at the time of transmission of a track report, the local TQ is compared with the last received remote TQ and a determination made to transmit or not to transmit the track report using the following rules:

1. The first unit to establish a track report has $R^2$ for that track.

2. A unit reporting a track is presumed to have $R^2$ for that track. A unit with $R^2$ for a real-time track shall relinquish $R^2$ upon receiving a remote real-time track report for that track.

3. A unit assumes $R^2$ on a common track if its local TQ at time of transmission exceeds the remote TQ by 2 when the locally held remote TQ value is 1 to 5. When the locally held remote TQ is 6, a unit may assume $R^2$ when its local TQ is 7.

4. A unit assumes $R^2$ if it has real-time data and non-real-time data was received.

5. A unit assumes $R^2$ if it has not received a remote report on a local track for approximately 40 seconds, or has decremented remote TQ such that item 3 applies, whichever occurs first. For determining $R^2$, if a remote report is not received, systems may decrement remote TQ by 1 approximately every 20 seconds until the remote TQ=1. TQ=0 shall not be assigned by decrementing.

6. A unit receiving a Link Drop track message on a locally held track shall assume $R^2$, if eligibility remains, at the next opportunity to transmit the track.

7. A unit with responsibility for reporting a track retains the responsibility until relinquished in accordance with the above rules or until the track is dropped. The $R^2$ unit shall transmit the Link Drop track message when a track is dropped.

8. A unit relinquishing $R^2$ on a track in accordance with the above rules shall not transmit the Link Drop Track message on that track.

### *Track reporting rules*

The Link-11 PUs implement the following track reporting rules:

1. Established local air or surface tracks that cannot be correlated with a remote track being received by the detecting unit shall be reported with all known parameters at the next transmission opportunity.

2. A unit reporting a real-time track shall report the position of the track extrapolated to the time of transmission.

3. A Forwarding Participating Unit (FPU) or Forwarding Reporting Unit (FRU) forwarding a real-time track shall not extrapolate that track position to the time of transmission.

4. Non-real-time track position shall not be extrapolated.

5. The air or surface track position message is sent on each real-time track (TQ=1–7) by the unit holding $R^2$ at the first transmission opportunity. Thereafter it need not be sent more frequently than every 8 seconds, but should be sent at intervals no greater than 20 seconds.

6. The air or surface track position message is transmitted upon receipt of information differences in Identity or Primary Identity (PRI) Amplification in the Information Difference Report or upon a manually initiated change in ID or PRI Amp fields.

### 3.1.2 Link-16

Link-16 uses the principle of Time Division Multiple Access (TDMA) protocol, which uses time interlacing to provide multiple and apparently simultaneous communication nets. A unit participating in Link-16 is assigned either to transmit or to receive in each time slot (each time slot is 1/128 second).

In Link-16, either 3, 6 or 12 Link-16 words can be transmitted in a 1/128 second time slot, depending on the data packing structure used: Standard, Packed-2 or Packed-4. A Link-16 word is analogous to a Link-11 frame. Each Link-16 word comprises 70 bits of data. A Link-16 message is composed of a variable number of words (normally 1, 2 or 3) although messages longer than 40 words are possible. Link-16 supports three types of messages: fixed format, free text and variable format. The Link-16 fixed format used to exchange tactical information is known as the J-series messages.

The Link-16 architecture is nodeless, i.e., does not require a node or unit to maintain communication. In Link-11, for example, the NCS is a node; if the NCS goes down, the network goes down. Link-16 does not need a dedicated station to maintain the network. When the Link-16 network is established, a single participating unit transmits a Network Time Reference (NTR) to establish the network system time. All other units use the NTR message to synchronize with the network. Once the NTR and the network have been established, the Link-16 network can continue to operate regardless of the participation of any particular unit.

Link-16 employs a five-character alphanumeric (19 bits) track number (TN) within the range 00001–77777 or within the range 0A000–ZZ777, allowing up to 524,284 TNs. Link-16 operates in a Track Block system because of its TDMA architecture, which does not permit proper TN accountability and therefore prevents Link-16 from operating in a TN pool mode.

The TQ value used by Link-16 relates to the accuracy of the reported position of the track.

In Link-16 TQ is a numerical value from 0 to 15. A specific positional accuracy defines each TQ value, except for a TQ of 0, which defines a non real-time track. The highest Link-16 TQ value requires better than 50-foot accuracy.

### 3.1.3 Link-22

The architecture employed in Link-22 can be either TDMA or Dynamic TDMA (DTDMA) and therefore is also considered a nodeless network architecture. Link-22 is a "Link-16 Family" datalink, along with Satellite Tactical Data Link (STDL), Satellite Tactical Data Information Link J (S-TADIL J), Link-16 and Variable Message Format (VMF). Each Link-22 word comprises 72 bits of data. As such the 72-bit word message standard can carry embedded J-series Link-16 messages, known as the FJ series, as well as newly defined Link-22 F-series messages.

#### *TQ assignment*

The TQ value used by Link-22 is a measure of positional information reliability.

The value 0 indicates a non-real-time report. Values 1 to 15 indicate different degrees of reliability of the positional data, 15 being the most reliable. The positional accuracy associated with each TQ value is defined as the area in square data miles within which there is a 95% probability that the track is actually located at the time of the report.

#### *Track $R^2$*

The following rules are designed to ensure that only the NILE Unit (NU) having the best positional data available is reporting the track on the interface:

1. The first NU to report on the track has $R^2$ that track.

2. An NU transmits a track report for an air, surface or land track only when it has $R^2$ for that track.

3. An NU assumes $R^2$ on a common track if its local TQ at time of transmission exceeds the received TQ by 2 or more, except when TQ is equal to 15. In this case, the NU with a TQ of 15 may assume $R^2$ for a track being reported with a TQ equal to 14.

4. An NU assumes $R^2$ if it has local real-time data and non-real-time data were received.

5. For standard update rate periodic track reporting, an NU assumes $R^2$ if it has not received a remote report on a locally held space track for approximately 25 seconds, an air track for approximately 40 seconds, or a locally held surface or land track for approximately 240 seconds.

6. An NU receiving a drop track message for a locally held track shall assume $R^2$ at the next opportunity to transmit the track if local reporting eligibility remains and a remote report has not been received.

7. An NU without $R^2$ for a non-real-time track with TQ field set to value 0 shall assume $R^2$ for that track when the track is locally updated by a new non-real-time report. An NU reporting a non-real-time track has $R^2$ regardless of the time value in the track report. The time value in the track report is not a criterion for an $R^2$ shift.

8. An NU receiving a Participant Location and Identification (PLI) message with the NPS Indicator set to Inactive, Conditional Radio Silence, or Tactical Data System (TDS) failure from an NU that has $R^2$ for a locally held track may assume $R^2$ at the next opportunity to transmit the track if local reporting eligibility remains and a remote report has not been received on the track.

9. An NU relinquishes $R^2$ for a real-time track upon receipt of a remote track report in which the report TQ is greater than the local TQ, or upon reception of a remote track report for the track which contains a TQ equal to the local TQ from an NU whose source TN is greater than local source TN.

10. An NU with $R^2$ on a track retains the responsibility until it is relinquished in accordance with the above rules or until the track is dropped.

11. The drop track message is transmitted when an NU ceases reporting a track for which it currently has $R^2$, except in the following cases:

    a. When an NU ceases reporting a track because it has relinquished $R^2$ to another NU

    b. When that track becomes an active NU on the network with the same TN.

### *Track reporting rules*

The Link-22 NU implements the following track reporting rules:

1. The System Network Controller shall provide the source of track data (NILE address).

2. Established local tracks that cannot be correlated with a remote track being received by the detecting NU shall be reported with all known parameters at the next reporting opportunity.

3. An NU reporting a real-time track shall report the position of the track extrapolated to the time of transmission.

4. Non-real-time data shall be identified as such and time tags shall be provided. Non-real-time track position shall not be extrapolated.

## 3.2 Generic concepts for distributed tactical picture

The generation of a tactical picture can be either centralized or distributed. The fully centralized approach requires that one of the units in a task group generate the tactical picture for the group. It is the role of the tactical centre to push information, in this case to perform multi-cast of the tactical picture information, to all units that are linked to the network. Other units (e.g., participants on the net) will push information collected from their local sources to the tactical centre. The centralized system ensures that all participants on the net have a single consolidated view. The major drawbacks of such systems are the lack of survivability of the participating units, and the large volume of data transferred on the network. Also an alternate tactical centre may be required to incorporate redundancy for the information distribution network.

At the other end of the architecture spectrum, the distributed approach requires that all participants broadcast all local data available. Each unit in the task group would then generate its own tactical picture

of its respective data sources with the data supplied by other units in the task group.  Each unit linked to the network collects, compiles and correlates the information.

The volume of data transferred on the network could be further reduced by the implementation of a hybrid version of the distributed approach, where each unit coordinates which local information is to be transferred, based on reporting responsibility.

# 4. Test-bed high-level requirements

The test-bed shown in Figure 8 is composed of various applications communicating via socket connections. The System Manager application monitors and controls the complete test-bed. The STIM Driver application reads input simulation data and sends it to other applications of the test-bed. In particular, it sends navigational and sensor data to MSDF of each platform. The Net Manager application is responsible for communications between platforms, and emulates link polling of PUs. The Run-Time Display (RTD) application is a research graphical user interface displaying, amongst other information, the tactical picture computed by the platform's Command and Control Information System (CCIS). The Concept Analysis and Simulation Environment for Automatic Target Tracking and Identification (CASE-ATTI) system developed at DRDC Valcartier is a high-fidelity simulator that emulates the behaviour of real targets, sensor systems, and the meteorological environment of the Canadian Patrol Frigate (CPF). The CloseLoopConcentrator application manages the messages exchanged between one or many CCIS applications and the CASE-ATTI application.

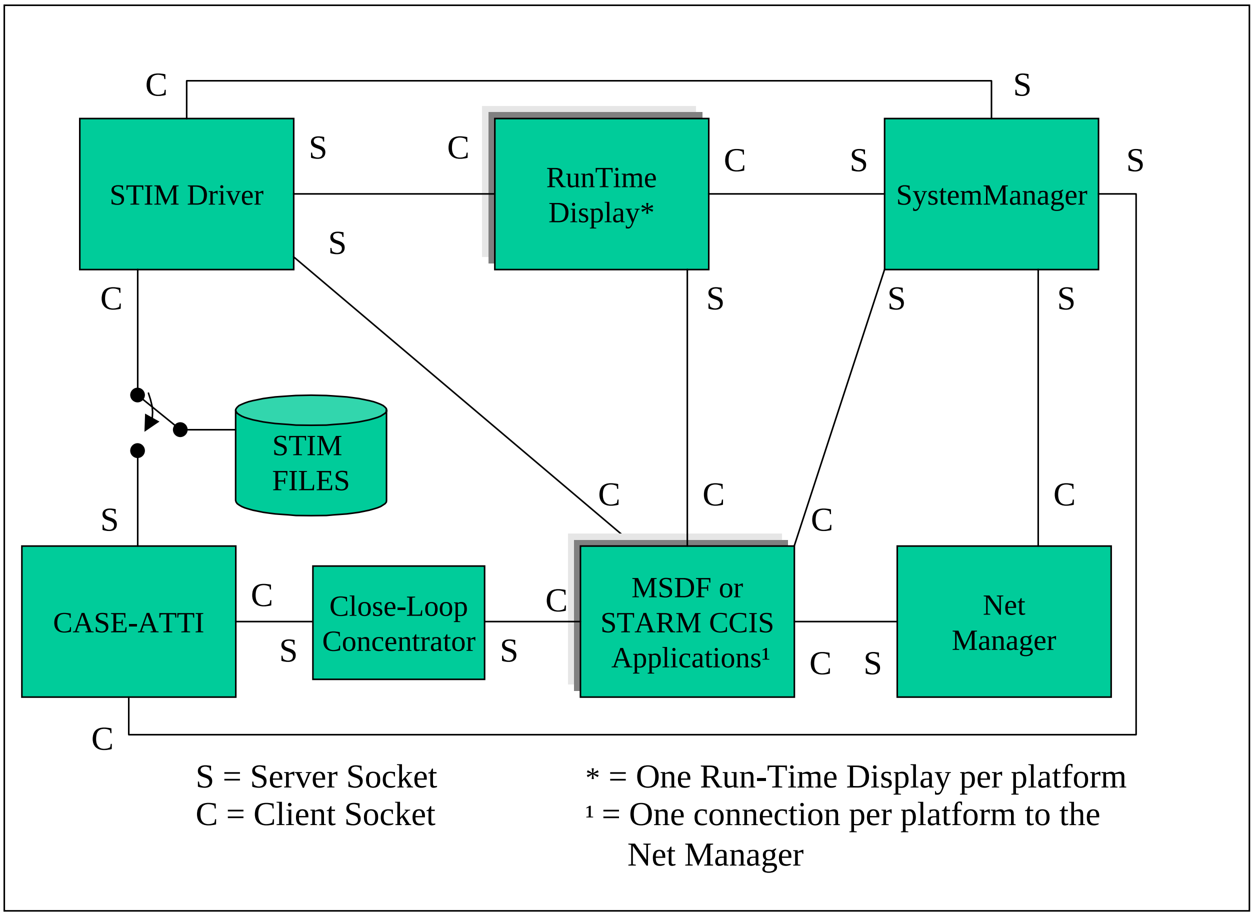



*Figure 8. Multi-platform R&D test-bed*

The test-bed can implement the distributed tactical picture approach, where each platform fuses its own local information with the information received from other PUs (via a link-like implementation scheme) or different data sources.

Currently, the agent-based MSDF is implemented to perform level-1 contact fusion using local sensor information. It does not fuse information received from other platforms participating in a link-like capability.

The multi-platform test-bed is enhanced to provide the infrastructure to support algorithmic development, communication exchange between CCIS, and agent-based approaches for rule-based information management implementation. The local MSDF capability will maintain track information pertaining to the LAP. The global MSDF capability will maintain the Global Tactical Picture (GTP) by the merging and fusion of the LAP and a real/near-real time Tactical Data Link (TDL) picture (e.g., Link-11). Future development of the Global MSDF will include the merging and fusion of the WAP. The Information Management capability will include requirements pertaining to the network protocol and filters capability. The Information Management capability will be implemented in an agent-based approach (BB under Cortex).

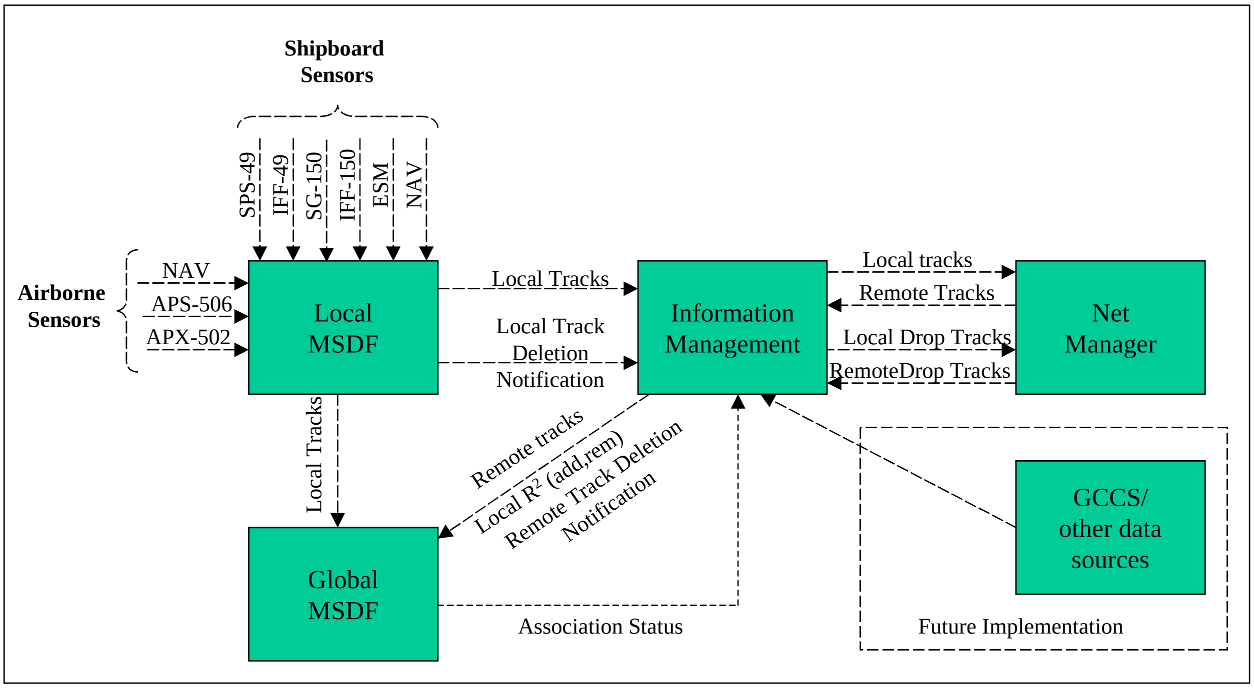



*Figure 9. High-level data flow for multi-platform data exchange*

Figure 9 illustrates high-level data flow between the various CCIS components of a platform and the Net Manager application.

The Information Management capabilities include:

1. Registering the platform with the Net Manager as a participating unit of the link-like implementation,

2. Link track number allocation for a local track,

3. Computing and managing TQ for a local track,

4. Reporting local track information to the Net Manager based on $R^2$ rules,

5. Receiving remote track information from other PUs and forwarding it to the Global MSDF,

6. Managing TQ for remote tracks,

7. Managing $R^2$,

8. Supporting Inputs/Outputs track information filters (e.g., allegiance).

The Global MSDF capabilities include:

1. Accepting local track updates from Local MSDF,

2. Accepting remote track updates from Information Management,

3. Performing local-to-remote and remote-to-local track correlation (time and position alignment) and association,

4. Performing positional fusion (if required),

5. Performing ID fusion (if required),

6. Managing Track deletion of global tracks.

The Association Status indicates whether a local track has been associated with a remote track (values: Unknown, Not Associated to remote, Associated). Upon creation of a local track, an Association Status of Unknown should be assigned. The Global MSDF will need to be notified when a unit has $R^2$ or has relinquished $R^2$ for a local track in order to determine if the data should be overwritten/fused (this is dependent on the value assigned to the position_fusion_mode parameter defined in the Global MSDF parameter file) by remote track data associated with the local track.

In the test-bed the Net Manager assumes the NCS functionality. The Net Manager sequentially polls each PU registered as a client application for track data information. The Net Manager application then dispatches the track data information received from a PU platform to the other PUs in the network. The Net Manager will poll a given PU at a rate (Polling Cycle Time) specified in the NetManager.config file.

The test-bed will initially support two modes of reporting local information to other platforms:

1. Report all local information
2. Report local information based on track $R^2$.

Since all platforms participating in the net should adhere to a common data exchange protocol, a parameter in the “CCIS Parameter File Elements” shall define the net reporting mode (i.e., all local tracks or local tracks based on $R^2$). This parameter will be used by the Information Management and Global MSDF applications.

In the test-bed, the $R^2$ will be tied to the positional and kinematic information of a track. This involves the test-bed supporting the investigation of having more than one unit reporting identity information about a track.

When running in multi-platform mode and performing data exchange via the Net Manager, the test-bed will have to run in real time. When connected to the CASE-ATTI application, which outputs the data as

fast as possible, the STIM Driver will manage (send) the data to the appropriate CCIS application based on message data stamp and simulation time.

## 4.1 TN assignment

In the test-bed, the track block system will be used. Currently the system uses the following equation to assign a track number: Link Track Number = 1000 + (reporting_platform_id)*100 + number_of_local_track_reported.

Once a TN has been assigned for reporting tactical information, that TN will be associated with that track data for as long as the data are reported regardless of which unit is reporting the data.

## 4.2 TQ assignment

The test-bed does not currently support non-real-time tracks.

In the test-bed the TQ system will use numerical values ranging from 0 to 15 (as for NILE). The value 0 is reserved for non-real-time reports. Values 1 to 15 will indicate different degrees of reliability of the positional data, where 15 represent the highest track quality. TQ will be based on the uncertainty area derived from the track positional $\sigma$ values.

A specific positional accuracy range (in $km^2$) will define each TQ value. There may be a need to define two sets of TQ values: one for surface tracks and one for air tracks. For a given TQ value, surface tracks should have a smaller positional accuracy range than air tracks. All platforms participating in the net should use the same TQ values and associated positional accuracy ranges, therefore these will be defined in the Information Management Parameter file (see Table 2). The air_tq_*n*_upper_limit is expressed in $km^2$.

*Table 2. Information management parameter file TQ definitions*

| TQ Value | Air Track | Surface Track |
|---|---|---|
| 0 | Non-Real Time Track | Non-Real Time Track |
| 1 | > air_tq_2_upper_limit | > surface_tq_2_upper_limit |
| 2 | > air_tq_3_upper_limit to air_tq_2_upper_limit | > surface_tq_3_upper_limit to surface_tq_2_upper_limit |
| 3 | > air_tq_4_upper_limit to air_tq_3_upper_limit | > surface_tq_4_upper_limit to surface_tq_3_upper_limit |
| 4 | > air_tq_5_upper_limit to air_tq_4_upper_limit | > surface_tq_5_upper_limit to surface_tq_4_upper_limit |
| 5 | > air_tq_6_upper_limit to air_tq_5_upper_limit | > surface_tq_6_upper_limit to surface_tq_5_upper_limit |
| 6 | > air_tq_7_upper_limit to air_tq_6_upper_limit | > surface_tq_7_upper_limit to surface_tq_6_upper_limit |
| 7 | > air_tq_8_upper_limit to | > surface_tq_8_upper_limit to |

| TQ Value | Air Track | Surface Track |
|---|---|---|
| | air_tq_7_upper_limit | surface_tq_7_upper_limit |
| 8 | > air_tq_9_upper_limit to air_tq_8_upper_limit | > surface_tq_9_upper_limit to surface_tq_8_upper_limit |
| 9 | > air_tq_10_upper_limit to air_tq_9_upper_limit | > surface_tq_10_upper_limit to surface_tq_9_upper_limit |
| 10 | > air_tq_11_upper_limit to air_tq_10_upper_limit | > surface_tq_11_upper_limit to surface_tq_10_upper_limit |
| 11 | > air_tq_12_upper_limit to air_tq_11_upper_limit | > surface_tq_12_upper_limit to surface_tq_11_upper_limit |
| 12 | > air_tq_13_upper_limit to air_tq_12_upper_limit | > surface_tq_13_upper_limit to surface_tq_12_upper_limit |
| 13 | > air_tq_14_upper_limit to air_tq_13_upper_limit | > surface_tq_14_upper_limit to surface_tq_13_upper_limit |
| 14 | > air_tq_15_upper_limit to air_tq_14_upper_limit | > surface_tq_15_upper_limit to surface_tq_14_upper_limit |
| 15 | 0 to air_tq_15_upper_limit | 0 to surface_tq_15_upper_limit |

## 4.3 Track $R^2$

A unit will assume $R^2$ for established local air or surface tracks that cannot be correlated with a remote track being received by the detecting unit.

For local air or surface tracks that have been or are correlated with a remote track, the unit will, at the time of transmission of the track report, compare the local TQ and remote TQ and make a determination made to transmit or not to transmit the track report using the following rules:

1. A unit reporting a track is presumed to have $R^2$ for that track. A unit with $R^2$ for a real-time track shall relinquish $R^2$ upon receiving a remote real-time track report for that track.

2. At every transmission opportunity, to determine $R^2$ the unit will, if a remote report has not been received within a time interval specified in the Information Management Parameter file (tq_decrementation_time_interval), decrement the remote TQ by 1 until the remote TQ=1. TQ=0 shall not be assigned by decrementing.

3. A unit assumes $R^2$ on a common track if its local TQ at time of transmission exceeds the remote TQ by a value delta_tq_r2_criteria (implemented as a parameter defined in the Information Management Parameter file) when the locally held remote TQ value is 1 to 13. When the locally held remote TQ is 14, a unit may assume $R^2$ when its local TQ is 15.

4. A unit receiving a Link Drop track message for a locally held track shall assume $R^2$, if eligibility remains, at the next opportunity to transmit the track.

5. A unit with responsibility for reporting a track retains the responsibility until relinquished in accordance with the above rules or until the track is dropped. The $R^2$ unit shall transmit the Link Drop track message when a track is dropped.

6. A unit relinquishing $R^2$ on a track in accordance with the above rules shall not transmit the Link Drop Track message for that track.

## 4.4 Track reporting rules

The test-bed can implement a mechanism by which a unit can report local tracks based on track $R^2$.

In the case where a local track needs to be reported based on $R^2$, the following track reporting rules will apply:

1. At every transmission opportunity, a unit will compute the TQ and send a track message for each local air real-time or surface real-time track (TQ=1–15) that has an Association Status ≠ Unknown and for which the unit holds $R^2$.

2. A unit reporting a real-time track (i.e., the Information Management application) will report the position of the local track at the time of the last update received from the Local MSDF fusion engine. **N.B.** This implementation differs from the Link-11 implementation and the proposed approach described in Section 2.4.4, where a unit reporting a real-time track reports the position of the track extrapolated to the time of transmission.

In the case where local tracks are not to be reported based on $R^2$ but all local tracks are reported, the following track reporting rules will apply:

1. At every transmission opportunity, a unit will compute the TQ and send a track message for each local air real-time or surface real-time track (TQ=1–15).

2. A unit reporting a real-time track (i.e., the Information Management application) will report the position of the local track at the time of the last update received from the Local MSDF fusion engine. **N.B.** This implementation differs from the Link-11 implementation and the proposed approach described in Section 2.4.4, where a unit reporting a real-time track reports the position of the track extrapolated to the time of transmission.

## 4.5 Data exchange

The test-bed should allow investigation of track-to-track (positional and ID) fusion, but by the same token it must allow the investigation of solutions for track-to-track fusion taking into account the current capabilities offered by the existing Link-11 system. Table 3 provides a list of track data information and an indication if they are supported by Link-11 messaging or in the link test-bed message.

*Table 3. Link air and surface track report content*

| **Track information** | **Link-11 messages** | **Net track data message in test-bed** |
|---|---|---|
| Category | √ [1] | √ |
| Link Track Number | √ | √ |
| Identity | √ | √ (allegiance) |
| Primary Identity Amplification | √ | |
| X Position | √ | √ |
| Y Position | √ | √ |
| Height/Altitude | √ [2] | √ |
| Height Source | √ [2] | |
| X Velocity | √ [2] | √ |
| Y Velocity | √ [2] | √ |
| Time | 2 & [3] | √ |
| Track Quality | √ | √ |
| Identity Amplification | √ [2] | |
| Covariance Matrix (upper triangle) | | √ |
| Specific Propositions | | √ |
| Subcategory | | √ |

In Link-11 data associated with Electronic Support Measures (ESM) FIX or ESM bearing are sent via M6B Link Messages. Table 4 provides a list of ESM data information and an indication if they are supported by Link-11 messaging or in the link test-bed message.

*Table 4. Link ESM report content*

| **ESM information** | **Link-11 messages** | **Net track data message in test-bed** |
|---|---|---|
| Track Type | √ [4] | √ |
| Link Track Number | √ | √ |
| Threat Evaluation | √ | |
| Type of Platform | √ | |

[1] Category is related to the Link-11 message type i.e., M2 = air, M3 = surface.

[2] These fields are contained in the Link-11 (M-8x) Amplification Messages.

[3] In M2 and M3 messages the reported time is used only for non-real-time tracks. In Link-11 the track's position is extrapolated to the time of transmission.

[4] The M6B message contains an indicator stating whether it is a track (FIX) or a bearing. If a bearing is reported the x and y position is the origin of the bearing.

| **ESM information** | **Link-11 messages** | **Net track data message in test-bed** |
|---|---|---|
| X Position | √ | √ |
| Y Position | √ | √ |
| Time | √ [5] [6] | √ |
| Report Source | √ [5] | |
| Bearing Accuracy | √ [5] | √ (σ bearing) |
| Bearing | √ [5] | √ |
| Platform Evaluation Confidence | √ [5] | |
| Lock-on/SPY | √ [5] | |
| Frequency/Frequency Range | √ [5] | |
| Broad Classification | √ [5] | |
| Amplify Characteristic | √ [5] | |
| Emitter Number | √ [5] | |
| Mode Number | √ [5] | |
| Confidence | √ [5] | |
| Bearing Rate | | √ |
| σ Bearing Rate | | √ |
| Altitude | | √ |
| Allegiance | | √ |
| Category | | √ |
| Subcategory | | √ |
| Specific Propositions | | √ |

The net track data message will contain data that will support proper investigation of possible solutions for track positional and identity fusion.

[5] These fields are contained in the Link-11 (M-8x) Amplification Messages.

[6] Does not apply to real-time tracks or bearings. Indicates the time since the interception was entered or since the bearing was updated.

# 5. Implementation within a KBS blackboard system

The implementation of track-to-track fusion must allow investigations of the various algorithms and approaches described in Section 2. Moreover, investigations require specific multi-platform scenarios and the design and implementation of MOPs.

The selected approach for the implementation of the track-level fusion system is based on a powerful real-time KBS based on the BB paradigm, which supports real-time distributed processing. This KBS BB, called Cortex, was jointly developed by LM Canada and DRDC Valcartier. It allows multiple blackboards to run on multiple machines with integrated communications. These features permit a highly configurable and flexible design for track-level fusion.

## 5.1 Architecture overview

A three-blackboard architecture is chosen for a CCIS that fuses data from both local sensors and remote systems. Each blackboard has a specific function:

1. The first blackboard fuses local sensor data
2. The second blackboard fuses local tracks with remote tracks (link and GCCS)
3. The third blackboard manages information transfer from and to other PUs in the network.

All functions listed above may be grouped on a single blackboard, but separating them into multiple blackboards has the advantage of making the design more flexible, although some redundancy of data structures is introduced. With multiple blackboards, the following configurations are available:

1. Blackboards run in a single process on a single machine
2. Blackboards run in different processes on a single machine
3. Blackboards run in different processes on different machines.

These various configurations are easily reached by modifying the Cortex configuration file without having to recompile the system. Then, analysis of performance may be conducted to determine the optimal configuration with respect to Central Processing Unit (CPU) and network usage. Note that in a first implementation, only the single process configuration is available.

The three-blackboard architecture is depicted in Figure 10, including relationships with other applications of the test-bed. The three blackboards shown have the following purposes:

1. Local MSDF: Performs fusion of local sensor data, generating a LAP
2. Global MSDF: Performs fusion of the LAP with remote tracks, generating an integrated GTP
3. Information Management: Manages information transfer from and to other platforms and systems

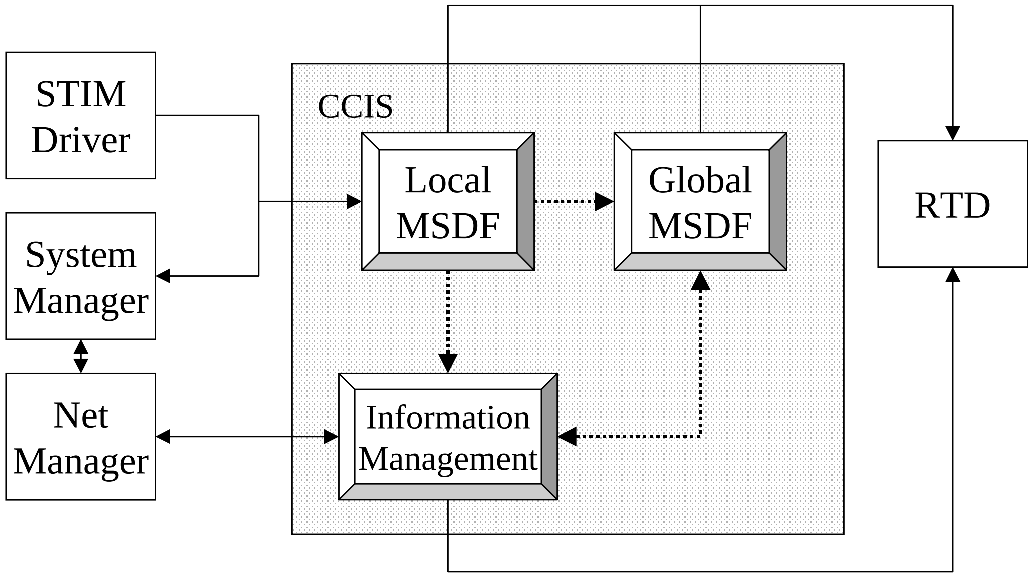



*Figure 10. Multiple blackboard CCIS in the test-bed*

Each of these blackboards is described in the following sections. In the above figure, dotted lines represent blackboard communication links, that is, connections that allow an agent running on one blackboard to put data on another one. For example, an agent on the Local MSDF can put data on the Global MSDF, similar to putting data on its own blackboard. Cortex takes care of the transmission if the other blackboard is remote, i.e., it is not part of the same process. The arrows describe the data flow between blackboards. For example, Information Management and Global MSDF put data on each other's blackboards, while Global MSDF does not put data on the Local MSDF blackboard, as the GTP should not interfere with the LAP.

Solid lines represent standard socket connections through which messages from the test-bed Messaging library are passed. These connections are used for communications with other test-bed applications like the STIM Driver, the RTD and the Net Manager.

### 5.1.1 Local MSDF

The Local MSDF blackboard fuses local sensor data to produce the LAP, which is composed of local tracks. It performs the following standard fusion processes for a single platform, as was described in a series of 3 DRDC-V reports for the CP-140 Aurora aircraft (TM-2004-281, TR-2004-282 and TR-2004-283). The Local MSDF must process data in real time while supporting a given number of local tracks and input data rate.

The same functions have also to be implemented on the HALIFAX class frigates, but with different algorithms, since the frigates are mostly concerned with fast aircraft and the Aurora investigates mostly the ID of slow-moving surface ships:

1. Data alignment

2. Positional and identity gating

3. Data association

4. Position and identification fusion

5. Track management.

Currently supported sensors are the long-range radar SPS-49 and the medium range radar SG-150 on the HALIFAX class frigates, the APS-506 on the Aurora, the TPS-59, and Identification Friend or Foe (IFF) slaved to each radar, as well as Electronic Support Measures (ESM). The main algorithms currently implemented are:

1. $\chi^2$ statistical distance for positional gating

2. Nearest Neighbour and Jonker-Volgenant-Castanon (JVC) single-scan methods and Multi-Hypotheses Tracking (MHT) for multi-scan association. The MHT was adapted from CASE-ATTI and required different track management and should be used when single-scan associators perform poorly, such as in dense target environments

3. Extended Adaptive Kalman Filters (EAKF) in 1-D, 2-D and 3-D, and a Constant-Velocity Constant-Acceleration (CVCA) Interacting Multiple Model (IMM) in 2-D and 3-D for positional fusion

4. Truncated Dempster-Shafer for identity gating and identity fusion

Some crude adaptive fusion is possible using the RTD provided to the operator. Adaptive fusion can be defined as a means to modify the behaviour and configuration of data fusion processes at run time. For example, one association or tracking algorithm may be more appropriate than another at a given time or in a given region. Adaptive fusion relies on a decision mechanism to modify MSDF behaviour at run time. This mechanism monitors MSDF and the behaviour of tracks to determine the best algorithms to use, the best set of parameters to configure them, etc.

In this way, the RTD allows the user to choose the association algorithms to be used for a given set of tracks. The RTD has the capability to display the associator currently assigned to tracks on the Tactical Situation Analysis (TSA). It also enables the user to display all tracks that have a given association, e.g., JVC. To change the associator for a set of tracks, the user selects the tracks and presses the Quick Action Button (QAB) for the desired associator. A command is then sent to the MSDF application to change the associator for those tracks. Special care must be taken in track management when switching from a single-scan to a multi-scan associator.

In addition to the MSDF functionality, there are also Situation and Threat Assessment (STA) and Resource Management (RM) applications for the frigates.

The STA/RM application is composed of two functional parts:

1. A basic STA/RM that performs a minimal set of STA and RM functionalities for a naval platform. The basic STA/RM performs three tasks:

   a. <u>Threat Evaluation</u>: This task consists of ranking all the non-friendly targets in the local picture according to their potential threat to the ownship. An ordered threat list is produced at the end of the process.

b. Engageability Computations: This task checks if any threat on the threat list can be engaged by the Surface-to-Air Missile (SAM) and gun weapons. Parameters such as the time of first fire and the time of intercept are computed for each threat.

c. Weapon Assignment. This task evaluates if any of the current weapons can be assigned to or engage either of the two highest threats.

2. An Advanced STA/RM that contains more exploratory algorithms. The Advanced STA/RM is composed of the following algorithms:

    a. Identity Refinement: These algorithms examine target behaviours to refine the target identity provided by MSDF

    b. Clustering: These algorithms detect if groups of targets can be defined based on their proximity, relative velocity and intent (hostile, friendly, etc.)

    c. Deliberative Planning: These algorithms produce an engagement plan on request. This engagement plan is generally more optimal than the reactive plan activated by the Weapon Assignment task. The plan is then shown to the user, who may decide to execute the plan instead of the usual automatic reactive plan.

### 5.1.2 Global MSDF

The Global MSDF blackboard has to perform track-level fusion of local tracks with all types of remote track data (link, GCCS, etc). This fusion generates an integrated GTP composed of global tracks. Since track-level fusion and contact-level fusion are different in essence, Global MSDF may reuse some, but not all, algorithms of Local MSDF. Mainly, positional fusion cannot be performed with EAKF or IMM because of possible cross-correlation between global tracks and reported (local and remote) tracks. In fact, positional fusion may not be required all the time. Therefore, it will have to use different algorithms than those used by the Local MSDF. Some problems that Global MSDF has to deal with are:

a. Cross-correlation between tracks (both position and identity)

b. Most reported tracks will be delayed in time compared with local tracks

c. Variable track histories

d. Wrong associations due to relatively large uncertainties of remote track data.

In order to solve positional cross-correlation between tracks, selective position fusion was implemented.

Selective position fusion involves processing local and remote positional data differently to prevent data incest. Local positional data come from the Local MSDF, while remote data come from Net, Link-11, GCCS-Maritime (GCCS-M) or Bearing Intercept Fix (BIF) sources. Depending on the source, the position of a reported track may be fused to the global track's position, may simply update (replace) the position of the global track, or may be discarded.

Track position information from the Local MSDF is never fused into a global track since the tracking has already been performed by the Local MSDF. Fusing this information again will introduce cross-correlation and make the filter used overly confident. Therefore, for tracks from the Local MSDF, positional information is added to the track history only if it is in the future of the track.

Track positional information from Net, Link-11, GCCS-M, and BIF sources is fused with the user-selected tracker. Note that, again, the fusion occurs only if the reported track is in the future of the global track. If it is in the past of the global track, the position information is simply discarded.

Other methods to prevent positional data incest are the use of tracklets (see Section 2.2.1) and Covariance Intersection (see Section 2.2.2).

### 5.1.3 Information Management (IM)

The Information Management blackboard is responsible for communications with track reporting systems according to the rules described in Section 4. In the context of the test-bed, Information Management manages information from and to the Net Manager. Note that only local tracks are broadcast to other PUs to prevent data incest. Information Management performs the following tasks:

a. Processes requests for data coming from the Net Manager or from another PU's CCIS

b. Filters the input/output of data according to some activated filters

c. Passes information received on remote tracks to the Global MSDF blackboard

d. Determines $R^2$ based on local and remote track qualities.

### 5.1.4 CCIS extensions

The blackboard-based design can be extended to incorporate Situation and Threat Assessment and Resource Management (STA/RM). A first integration of STA/RM is performed as depicted in Figure 11. In this case, the STA/RM blackboard resides in a different process and receives its input from Local MSDF. Data is transferred with the use of messages selected from the Messaging library.

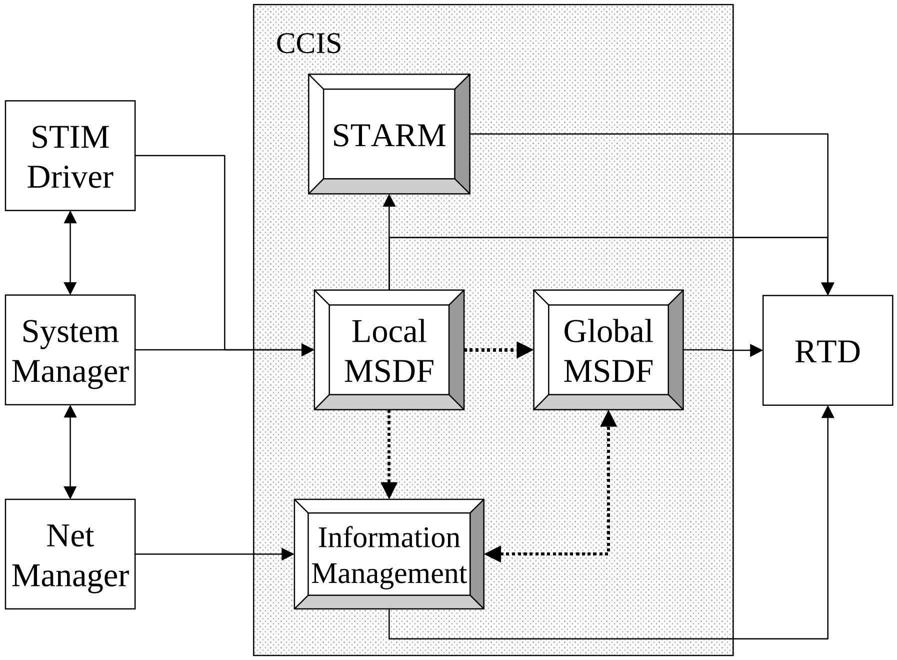



*Figure 11. Initial integration of STA/RM into multiple blackboard CCIS*

In the future, the blackboard-based design can be extended to incorporate Under Water Warfare (UWW) fusion and STA/RM, as depicted in Figure 12 for CPF sensors.

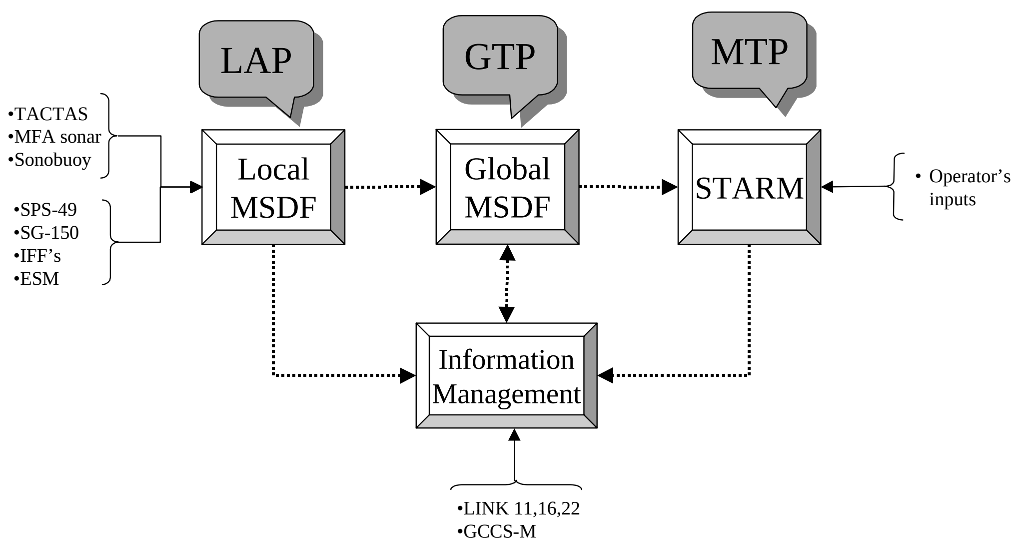



*Figure 12. CCIS with UWW data fusion and STA/RM*

The Local MSDF will have to be modified in order to correctly process UWW data, which may require the use of some track-level fusion algorithms implemented in Global MSDF. For example, track number gating may be used with UWW data, since UWW data typically contain tracks. In fact, UWW data may come from the module currently deployed on the CPF, or from another fusion engine working on UWW data only. Then, local tracks from both the AWW and UWW worlds, i.e., the LAP, may be passed to Information Management.

The STA/RM application will take the output of the Global MSDF, namely the GTP, as its input. Moreover, STA/RM will have to deal with operator inputs like manual track deletion and allegiance changes, which then form the integrated MTP. In principle, STA/RM should not feedback Global MSDF, as indicated by the unidirectional data flow between Global MSDF and STA/RM. Finally, STA/RM may send information to Information Management, especially during collaborative engagement scenarios.

## 5.2 Description of the three blackboards

This section presents a description of each blackboard: Local MSDF, Global MSDF, and Information Management. Issues regarding various aspects of each blackboard are presented and possible solutions are discussed.

### 5.2.1 Local MSDF

The Local MSDF blackboard performs the fusion of reports from platform sensors, which is currently performed by the MSDF application in the test-bed.

The Local MSDF blackboard contains data types and agents for the fusion of data from local sensors. It has agents to perform the following tasks:

a. Data alignment of sensors

b. Position and identification gating

c. Data association

d. Position and identification fusion

e. Track management.

Apart from data types used in the various fusion processes, Local MSDF should maintain one list of tracks resulting from the fusion of local sensor data. Note that in the future, UWW data fusion will be included and will require a track-level fusion algorithm.

When a track is created, updated or deleted, Local MSDF will distribute this information to interested blackboards, namely the Global MSDF and the Information Management blackboards. This is achieved through Cortex functions, i.e., data are simply put on remote blackboards and the controller takes care of the transmission. Instead of transmitting tracks individually, tracks are packed together for transmission, which makes it easier for Global MSDF to perform association and fusion.

Specifically, Local MSDF must pass the following positional information to both Global MSDF and Information Management:

a. State vector

b. Covariance matrix.

Note that no history should be passed, as that information will be built by the receiving applications and will be built differently than it is in Local MSDF. Also, by passing the covariance matrix, Local MSDF does not have to compute a track quality; this computation will be performed by Information

Management. Currently, it is anticipated that Global MSDF and IM will only need the position portion of the covariance matrix. Therefore, other components need not be passed at the moment.

Regarding identity information, Local MSDF passes the updated specific proposition set for each track. In the test-bed, it is also this information that is broadcast to other platforms by IM. Since this is binary information, it is mandatory that all platforms have their platform database synchronized.

The rate at which MSDF distributes its updated local tracks has an impact on the timeliness of the GTP generated by Global MSDF. In the current implementation, information on updated local tracks is sent to both Global MSDF and Information Management every time a buffer of contacts is processed by Local MSDF. Other options, like those described in paragraph 2.4.2, will be investigated in the future.

### 5.2.2 Global MSDF

The Global MSDF blackboard has to perform track-level fusion of local tracks with all types of remote track data (link, GCCS, etc). Therefore, the algorithms it uses are different from those used by Local MSDF for each of the fusion components:

1. data alignment
2. data association
3. position fusion
4. identification fusion

as discussed further in the following subsections.

#### *Data alignment*

Data alignment is composed of two tasks: spatial alignment of reported tracks and time alignment of selected global tracks.

Link-type data is usually in Cartesian coordinates (XY) relative to a Data Link Reference Point (DLRP). However, for distances greater than about 300-400 km, the effect of the earth's curvature is no longer negligible, and using a flat Cartesian coordinate system will introduce distortions. GCCS-M data may use longitude latitude, which is appropriate for global data. Global MSDF should use the longitude/latitude coordinate system for its tracking to take earth curvature into account. Therefore, an earth model should be selected in order to perform coordinate transformation between longitude/latitude and Cartesian.

For now, only Cartesian coordinates are supported since GCCS-M data is not available yet.

In order to compute a probability of association between reported and global tracks, each candidate track-track pair must be time-aligned. This is done by time-updating the global track to the time of the reported track. When the time difference between the reported and global tracks is positive, i.e., the reported track is in the future of the global track, this procedure is quite standard. The kinematic model is used to propagate the track's state vector and covariance matrix in the future.

On the other hand, when the time difference is negative, i.e., the reported track is in the past of the global track, different approaches are possible (see also Section 2.4.3). The simplest approach is to propagate

the global track back in time, as in the previous case, but this may introduce some errors since the global track may have performed a manoeuvre in its past and propagating with a constant velocity model would be incorrect. Another approach is to go into the global track's history and get a history point such that the time difference is positive.

This second approach requires that Global MSDF maintain long enough histories for possible time latencies of the system. For example, link tracks may be a few dozen seconds late, while GCCS-M tracks may be many hours late. Moreover, the length of the history kept for a track may depend on whether it is an air or surface track. In fact, the history of air tracks should be shorter than the history of surface tracks, since the former move much faster.

### *Data association*

In order to perform data association, Global MSDF should compute probabilities of association of incoming track reports with existing global tracks, based on the following types of information:

a. Position and velocity
b. Identity
c. Track numbers.

Remote Bearing Only (BO) tracks require special care during association because bearing intersections have to be correctly identified. The above topics are described in the following paragraphs.

Once a probability of association is computed for each reported track/global track candidate pair, a standard association algorithm like NN or JVC can be used to resolve the candidate pairs. Note that if reported tracks do not come in buffers, but individually, then an associator is almost useless. Therefore, it is preferable to have reported tracks packed into a buffer before they are processed by Global MSDF.

Instead of using a one-to-one association algorithm, one may want to investigate the use of an associator that associates more than one reported track with a global track. This approach may be useful in distributing the identity information of a BO track over highly possible global tracks, but will need further investigation in the future.

Positional gating is similar to the gating algorithms currently available in Local MSDF, that is, a statistical distance is computed between a reported track and a global track. Since velocity is available for reported tracks, a statistical distance computed with velocity as well as position can be used. The computed statistical distance can then be transformed into a probability of association for a system with four degrees of freedom (two-dimensional tracks).

When the positions of reported tracks do not follow a Gaussian distribution, a fuzzy logic positional gating approach may be investigated. This approach, called geo-feasibility, is currently used in the Adaptive Fuzzy Logic Correlator (AFLC). This may apply to some GCCS-M data, but may not be needed for link type data. Note that this type of association is not intended for implementation in the near future.

Identity gating may be performed the way it is now done by Local MSDF, which is related to the conflict $\kappa$ between the proposition sets of the reported and global tracks. The probability of association is then given by

$$P_{ID} = 1 - \kappa$$

Track numbers from remote reporting systems like link and GCCS-M may be used as attributes and used during gating. The heuristic is to increase the probability of association if a global track has been associated often with a remote track with a given track number in the past. The more often it has been associated in the past, the more probable the track/track candidate pair is. Note that track number gating will provide a factor for weighting the total probability of association computed from positional and identity gating. This way, when track number gating is not conclusive, the association will be based only on the previously computed total probability of association. Different approaches are available.

One approach is to compute the portion of track numbers of a given type (Local, link, GCCS-M, etc.). For example, let's say that a given global track "contains" 90% of link track number 4 and 10% of link track number 12 in its history. If the candidate reported link track has track number 4, then the probability of association should increase. On the other hand, if the candidate reported link track has track number 12 or any other value, the probability of association should either decrease or remain the same.

The value $\eta_{TN}$ is the proportion of reported tracks with track number TN in the "recent" history of a given track. How far back the track number history is scanned should depend on a user-configurable parameter. In the above example, we would obtain $\eta_4 = 0.9$ and $\eta_{12} = 0.1$ for link track numbers 4 and 12, respectively.

Once the above probabilities of association are computed, it is necessary to combine them into a total probability of association for the association algorithm. As in Local MSDF, positional and identity probabilities of association are combined with the following equation:

$$P_{tot} = P_{pos} P_{ID}$$

When there is no conflict between the two proposition sets, $P_{ID} = 1$ and the total probability of association equals the positional probability of association. Thus, the identity probability of association can only decrease the total probability of association.

Track number gating should increase or decrease the total probability of association, and therefore we cannot simply multiply by a given track number probability $P_{TN}$. The output of track number gating should be thought of as a weighting factor.

A weighted total probability of association is defined as

$$P_{tot}^{'} = P_{tot} + \Delta\eta_{TN} - P_{tot}\Delta\eta_{TN}$$

where

$$\Delta\eta_{TN} \equiv \eta_{TN} - \eta_{\min}$$

is the difference between $\eta_{TN}$ and the user-defined threshold $\eta_{\min}$. For a positive value of $\Delta\eta_{TN}$, the total probability of association is increased, while it is decreased for negative value of $\Delta\eta_{TN}$. The threshold indicates the ratio value above which the total probability of association should increase. Initially, a threshold of 0.5 may be used. Note also that $P_{tot}^{'}$ depends linearly on $\Delta\eta_{TN}$, such that the

larger $\Delta\eta_{TN}$ is, the more $P_{tot}^{'}$ will change relative to $P_{tot}$. The above equation ensures that $P_{tot}^{'} \leq 1.0$, but it may also be negative.

There is one special case of the above equation that warrants a closer look. When $\eta_{TN} = 1.0$ and $\eta_{\min} = 0.0$ the equation reduces to $P_{tot}^{'} = 1.0$, which does not depend on $P_{tot}$. That is, the total probability of association does not depend on positional and identity probabilities of association. This case should never happen, and the following should always be satisfied:

$$\eta_{\min} > 0.0$$

As stated above, the recommended value for initial investigations is 0.5.

Association of BO tracks may lead to the construction of a two-dimensional track, which is the intersection of the two bearing lines. The association of two different bearing lines is a difficult problem because of the many possible associations in a dense target environment. Moreover, reported BO tracks may associate with existing two-dimensional global tracks, and not only with existing BO or BIF global tracks.

BO tracks are fully supported in Global MSDF. That is, reported BO tracks are associated and fused with existing global tracks (BO, XY, or XYZ). When reported BO tracks are received by Global MSDF, a sample of global tracks (BO, XY or XYZ) are selected for candidate associations. Then, all possible pairs of reported and global tracks are created, and global tracks are time-updated for processing by the gating function.

The positional gating of reported BO tracks with global BO tracks is allowed only when tracks have a similar point of origin. When origins are too far apart, the two BO tracks cannot be associated with each other, and the probability of association is set to 0 to force this. When origins are similar, a one-dimensional $\chi^2$ statistical distance is computed between the bearing values of the two tracks and then transformed into a probability of association according to standard methods.

Of course, no restriction with respect to the origin is applied when gating a reported BO track with XY or XYZ global tracks. In this case, a bearing value is computed for the global track relative to the origin of the reported BO track. A one-dimensional $\chi^2$ statistical distance is then computed between the bearing values and used to compute the positional probability of association.

Identity gating is also performed. It is obtained by computing the Dempster-Shafer conflict between the proposition sets of the reported BO track and the global track. The total probability of association is then computed by multiplying together the Dempster-Shafer conflict and the positional probability of association (see Section 2.3).

Once reported BO tracks are associated with a given global track, position and identity fusion are performed. Currently implemented position fusion algorithms are Latest Report and Selective Position fusion (see Section 5.1.2). Note that no position fusion is performed when a reported BO track is associated with an XY or XYZ global track, since a reported BO track contains less accurate positional information than the global track. Fusion is not performed in this case in order to prevent introduction of noise in the position of the global track. On the other hand, position fusion is performed when a reported BO track is associated with a global BO track. When filtering is required, which is only available with the Selective Position fusion algorithm, the one-dimensional (bearing only) EAKF is used. Fusion of BO reported tracks with BO global tracks is not supported by the IMM filter.

A BO track is defined by a two-dimensional state vector (bearing, bearing rate) and a 2x2 covariance matrix. However, link message formatting does not permit all this information to be sent/received in standard link messages. The content of the message is limited to the following: the time of the broadcast, the identification of the reporting unit, the bearing of the target and an approximate value (called track quality) of the uncertainty of the bearing. The bearing uncertainty is provided in predefined ranges of error.

Since the BO information provided by a remote platform consists of an approximation of the actual output of its BO trackers, this information is used to enhance the local BO information collected by the receiving platform. As a consequence, the remote to global BO track-to-track fusion process is designed in such a way that the output of the local BO trackers will stay uncorrupted.

Ghosts are produced by multiple intersects from several bearing lines detected almost simultaneously by several PUs when trying to perform a BIF calculation. A specific strategy for ghost reduction must therefore be implemented. The validity of BO intersects must be tested (range to target must be in the detection range, emitters must be consistent with an existing target) before being validated as the potential location of a target. The range and identity of the target are used during validation. Once validated, the dynamic, i.e., velocity, of the BIF track could also be used for further ghost reduction. For example, a surface BIF track should not be allowed a speed of Mach 1. However, this type of constraint is difficult to enforce because although a ghost is actually removed, it may be regenerated later. This happens because dynamic information is obtained over many updates, and it cannot prevent the creation of a BIF track on a single BIF contact. Therefore, the use of dynamic conditions in ghost reduction requires more investigation.

BIF track processing is complex, but it can be decomposed into several steps. It is summarized here as a whole for convenience, even though some steps refer to positional fusion (step 6) and to track management (steps 7 and 8):

1. Processing at the reception of a remote BO track in the Global Track Data Base (GTDB)
2. Test if the remote BO track is already tracked
3. Test possible intersections of the remote BO track with existing global BO tracks
4. BIF contact creation
5. BIF contact to track gating
6. BIF track update
7. BIF track creation
8. BIF track deletion criteria

Step 1: Processing at the reception of a remote BO track in the Global Track Data Base (GTDB)

Information received from a remote platform is often, but not always, stale. If it is stale, the remote BO track and the global BO and XY tracks contained in the GTDB must be propagated over time. This is performed on a remote track/global track pair basis; that is, for each candidate track/track pair, the global track is time-updated to the time of the remote track.

Step 2: Test if the remote BO track is already tracked

1. Track selection: on receipt of a buffer of remote tracks containing BO and XY tracks, the coverage region of the buffer is computed. Then all BO and XY global tracks in this area are selected for potential association (gating) and update.

2. Association: the remote BO tracks are to be associated with one of the selected global tracks in the GTDB. For the reported BO track, the statistical distance between the bearing of the remote BO track and the bearing of the selected global track is computed. Finally, an assignment matrix is built containing probabilities of non-association for each track/track pair. This matrix is then resolved using either the NN or JVC algorithm.

Step 3: Test possible intersections of the remote BO track with existing global BO tracks

1. Track Selection: Once a global BO track is updated, a selection of other global BO tracks from the GTDB is performed by selecting all the global BO tracks from the GTDB having a bearing included in the angular interval defined by the bearing under which the remote platform is locally seen and the bearing of the reported remote BO track. Figure 13 shows the geometry of this process.

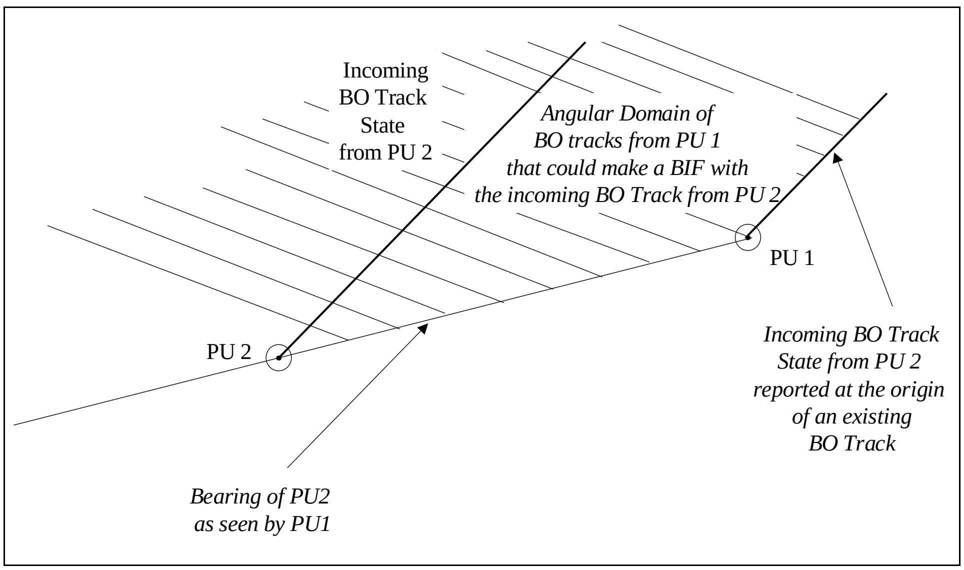



*Figure 13. Definition of the angular domain for the selection of BO tracks from the GTDB that would qualify for making a BIF with an incoming BO track from PU 2*

2. BIF validation: BIFs are computed between the updated global BO track and each of the selected global BO tracks. Two tests are performed to evaluate the feasibility that these BIF intersects may actually represent the location of a potential target. First, a range validation is performed as follows. The location of a potential target is estimated by calculating the range of the BIF from the two origins of the BO tracks that define it. If one of the estimated ranges exceeds a given threshold (representing the maximum range of detection of the passive sensor), then the BIF is declared out of range and is no longer considered as the location of a potential target. Figure 14 shows the geometry of the target range estimation. If a BIF location satisfies range validation, a second test verifies that the sets of target identity propositions associated with the BO tracks are not conflicting. If the Dempster-Shafer conflict between the two proposition sets exceeds a given threshold, then the BIF is invalidated.

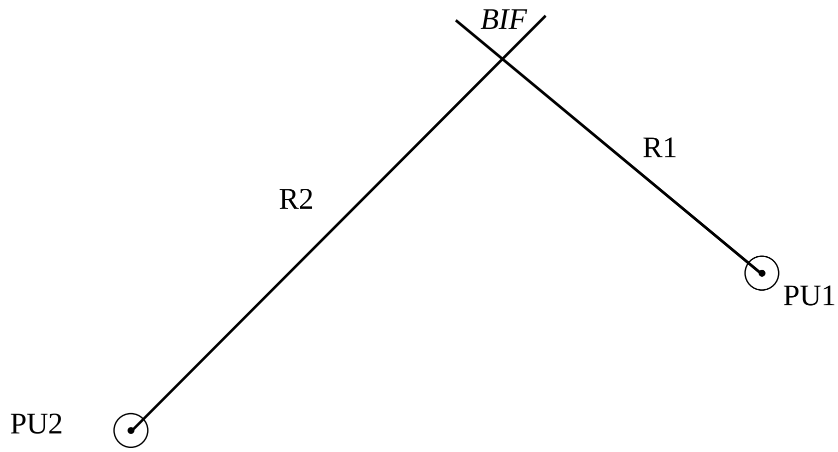


*Figure 14. Estimation of the range of a potential target located at a BIF intersect*

Step 4: BIF contact creation

Once a BIF has been successfully validated, a BIF contact is created. A BIF contact is created with an XY position and a covariance matrix like an XY contact, that is, with no velocity. BIF contacts are processed like any XY contact or track by the data fusion system. However, BIF contacts can only associate with and update existing BIF tracks in the GTDB.

The first step is to compute a state vector and a covariance matrix for the BIF. The state vector contains only position information and no velocity. The XY position of the BIF contact is the intersection of the two global BO tracks. The covariance matrix is computed using the area delimited by the intersection of the two triangular areas of uncertainty of the BO tracks (see Figure 15). The delimited area represents the uncertainty of the BIF contact. The covariance matrix is then obtained by converting this area into an ellipse enclosing all the vertices contained in the area.

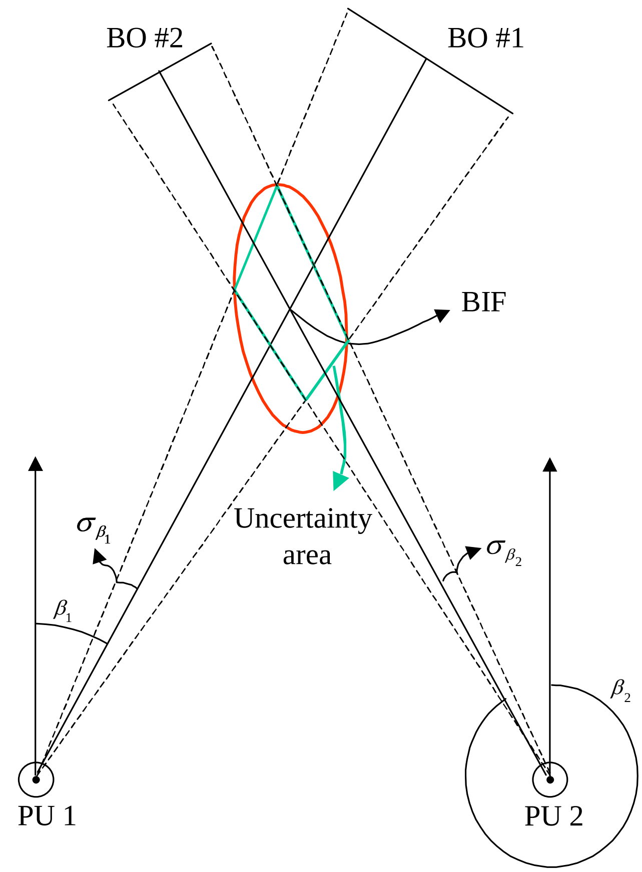



*Figure 15. Computation of the covariance matrix for a BIF contact*

The BIF track will track the valid BIFs in XY geometry and will be initiated as follows.

Step 5: BIF contact to track gating

The gating procedure of a BIF contact with a BIF track is identical to the gating of an XY contact with an XY track. A two-dimensional $\chi^2$ statistical distance is computed, which is then translated into a positional probability of association. A Dempster-Shafer conflict is computed between proposition sets of the BIF track and contact. The conflict and the positional probability of association are multiplied together and a total probability of association is obtained.

Next, an assignment matrix is built with the probability of non-association for each BIF contact/track pair. The matrix is then resolved with either NN or JVC. BIF contacts of confirmed pairs will update their associated BIF track.

Step 6: BIF track update

When a BIF contact is associated with a BIF track, the former is fused into the BIF track. This is performed using either the EAKF or the IMM filter, as it is for XY contact-to-track fusion.

Step 7: BIF track creation

When a BIF contact is not associated with a BIF track, a global BIF track is created. Since it is a two-dimensional contact, BIF track creation is similar to XY track creation. However, the new

track is identified as a BIF to make sure that reported BO tracks cannot associate with it in the future.

Step 8: BIF track deletion criteria

1. A BIF track may be removed from the GTDB if it has not been updated for some time. This first condition is tested during a periodic track management task. Periodically the GTDB is scanned to remove global tracks that have not been updated for a given time. In this case, BIF tracks are processed like other tracks in the GTDB.

2. A BIF track may be removed from the GTDB if it is confirmed by a reported XY track. This second condition is met when a reported XY track gets associated with a BIF track. Being confirmed, the BO tracks that generated this BIF are also removed from the GTDB; future reports of these BO tracks should be associated with the new global XY track. Moreover, any other BIF tracks that these BO tracks may have generated are also removed from the GTDB, because they are considered ghost BIF tracks.

3. A BIF track may be removed from the GTDB if one of its associated BO tracks was confirmed by a reported XY track. This third condition is met when a reported XY track gets associated with a global BO track. Then, any BIF tracks that this global BO track may have generated are removed from the GTDB, because they are now considered ghosts.

### *Position fusion*

Once a reported track and a global track are associated, their position and velocity are fused to provide an updated global track. Since cross-correlation exists between these two tracks, positional fusion must be performed with care. Different approaches exist for track-level positional fusion, and they are discussed in the following paragraphs. Once implemented, the user should be able to select one of them from a configuration file.

One approach to position fusion is simply to keep the most recent position update received for a global track, regardless of its origin. In this case, the operator will be refreshed with local data, although the current platform does not report its local track to other PUs. Since no fusion is actually performed, the problem of cross-correlation is circumvented.

In a network of PUs like the various flavours of link, only one PU is allowed to report data on a given track, that is, only one PU has $R^2$. In such a network, the problem of cross-correlation can be circumvented by not fusing positional data at all, and simply updating a global track with position and velocity provided by the PU having $R^2$. It is Information Management that decides whether or not the platform has $R^2$ for a given local track.

On the other hand, GCCS-M does not support the $R^2$ concept, so positional fusion may be required. However, GCCS-M data will typically have a sizable latency relative to local and link data, and positional fusion may not be required since more up-to-date information would already be available in Global MSDF.

Recall that a tracklet is a track built in such a way that its covariance does not incorporate cross-correlation with any other types of data (see Section 2.2.1 for a complete description of tracklets). Typically, a tracklet is computed from the last few measurements instead of incorporating all

measurements since the creation of the track. Instead of sending tracks to the Global MSDF, tracklets are passed between Local and Global MSDF. The latter fuses tracklets with existing global tracks.

Covariance intersection is a method that combines two tracks with unknown cross-correlation (see Section 2.2.2 for a complete description of Covariance Intersection). The two track state vectors and covariance matrices are combined through a convex combination, and the fused track ends up with a conservative covariance matrix. By conservative we mean that the error is not optimal, that is, it is larger than it should be.

#### *Identification fusion*

Global MSDF will use the same algorithm as Local MSDF, namely the truncated Dempster-Shafer algorithm. This algorithm takes the proposition set of the reported track and fuses it with the current proposition set of the global track. However, proposition sets from the same source should not be fused repeatedly because proposition sets from the global and reported tracks are cross-correlated. Fusing cross-correlated identity information will cause the global track identity to become overly confident in the identity given.

This situation is similar to repeatedly fusing the same information from a given sensor. For example, fusing the AIR proposition too many times will cause its mass to artificially increase and may make other information appear negligible. In Local MSDF, a “consecutiveness” criterion was introduced to prevent identical information from being fused repeatedly.

In Global MSDF, we could also adopt the consecutiveness approach, but this criterion is more difficult to define for global tracks because the identities of reported tracks (especially local tracks) are not formed as simple generic propositions. The other approach is to maintain as many proposition sets as there are sources of identity information for a global track. For example, a global track may maintain a proposition set from Local track and one from link track. Then, one other proposition set will be maintained, which contains the fusion of the various proposition sets. Proposition sets are fused together every time one of them is replaced with new data, thus producing a new fused proposition set.

This simple approach solves the problem of cross-correlation within identity.

The co-existence of the Dempster-Shafer approach—commonly used nowadays in the German F124 frigates, the Finnish Fast Attack Craft Squadron 2000, and in the Light Airborne Multi-Purpose System (LAMPS) helicopters of the US Navy—with the Bayesian approach proposed in STANAG 4162, both in internal fusion and in broadcasting that information, will be the subject of another separate report.

### 5.2.3 Information Management (IM)

The Information Management blackboard is responsible for the input and output of information to and from PUs and other systems. Its main tasks are management of:

a. Local tracks
b. Reporting responsibility and association
c. Reported tracks.

Therefore, it must work with Local MSDF, Global MSDF and the network (i.e., the Net Manager).

Information Management must work in two modes. First, in the reporting responsibility mode, it reports only local tracks for which it has reporting responsibility. In the second mode, it reports all its local tracks

irrespective of reporting responsibility. The following paragraphs describe the different tasks of Information Management.

### *Management of local tracks*

Information on created, modified and deleted local tracks is received from the Local MSDF blackboard. IM needs to store all this information in order to broadcast it upon request from the Net Manager. All last track reports sent by Local MSDF are stored in the local track list of IM.

Whenever a local track is deleted, a check is made to see if $R^2$ was set and, if so, a NetDropTrackMsg is sent to the network.

### *Management of reported tracks*

For TQ comparison purposes, Information Management must maintain a list of associations between local and remote tracks. The association is not performed by the IM, but by Global MSDF, and is therefore received from Global MSDF as soon as this information is ready. De-associations are taken into account by the same mechanism.

When polled by the Net Manager, the IM must scan its local list and decide whether or not a particular track should be broadcast. The decision is based on different filters applied sequentially. The filters can be applied on any attributes of the track, the most important being reporting responsibility and association status. When in $R^2$ mode, the IM will broadcast all tracks for which it already has $R^2$ and an association status different than unknown. For all other tracks, checks are made to see whether or not it should take over $R^2$. Those checks are described further below. When in report all mode, all local tracks are broadcast.

When a newly created local track is received, IM must wait until Global MSDF reports on its association status. If the local track is associated with an existing remote track, IM should compare track qualities in order to determine whether or not the current platform should take over reporting responsibility. If the local track is not associated with any remote tracks, IM assigns a new remote (link) track number and sends it to the Net Manager when requested. IM must monitor the TQ of reported and local tracks in order to take over $R^2$ whenever necessary. Rules defining this are described in Section 4.3.

Information Management must also maintain up-to-date information on the navigation data of the platform, as this information is also sent to other PUs at the beginning of every transmission in the form of a NetPUData message. This information will be periodically updated from Local MSDF.

### *Management of $R^2$*

At any time, IM may receive information from other PUs via the Net Manager. It can also receive information from other reporting systems like GCCS-M. Four kinds of messages can be received from the Net Manager: a NetPUDataMsg, a NetTrackDataMsg, a NetDropTrackMsg and a CommandMsg representing the end of data transmission. The NetPUDataMsg is received at the beginning of every transmission from the network, and essentially gives the reporting unit number for all following track reports. This message is processed by the agent ProcessNetPUDataMsg. The NetTrackDataMsg encapsulates all information about a remote track. Whenever a remote track is received, a check is made to see if there is a local track associated with it for which the PU has $R^2$. If this is the case, the $R^2$ is dropped.

## 5.3 Agent-based implementation

The following sections present the actual agent-based implementation on Cortex of Local MSDF, Global MSDF and IM. The agent-based implementation of Local MSDF was described in previous reports on single-platform data/information fusion on an airborne platform (Documents TM-2004-281, TR-2004-282 and TR-2004-283). Data flow diagrams are presented, which describe the flow of data between the agents on a given blackboard. In these diagrams, agents are grouped into functional blocks and arrows represent the flow of data between these blocks and between agents.

Next, a description of the main data types pertinent to each blackboard is presented. For each data type, a diagram presenting the relation between this data type and agents and other data types is shown. The symbols used in these diagrams are described in Table 5.

*Table 5. Description of symbols used in data type diagrams*

| | |
|---|---|
| DATA_TYPE | Represents a data type. |
| Agent name | Represents an agent. |
| ▭ Context function / Context function | Represents a context function. The attributes of the entrant data type (indicated by the ingoing arrow) are evaluated and the appropriate agent (one of the agents indicated by the outgoing arrows) is activated. The context function name is indicated at the right of the box. When multiple context functions are combined, they are separated by a slash (/). |
| ⟶ | Indicates a control or a data flow. When starting from a data type, it goes to the context function or the agent normally activated by the data. Otherwise it goes from the context function to the agent chosen by the context function. |
| ——————— | Indicates that the agent puts at least one new instance of the linked data type on the blackboard. |
| – – – – – – – – | Indicates that the agent modifies or reads at least one existing instance of the linked data type on the blackboard. |
| –·· –·· –·· –·· –··· | Indicates that the agent modifies or puts at least one instance of the linked data type on the blackboard. |

### 5.3.1 Local MSDF

The following presents the design of Local MSDF. The overall data flow is first presented in Figure 16. Then, each agent of the contact/track fusion functional block is presented in detail with a description of the different data types involved.

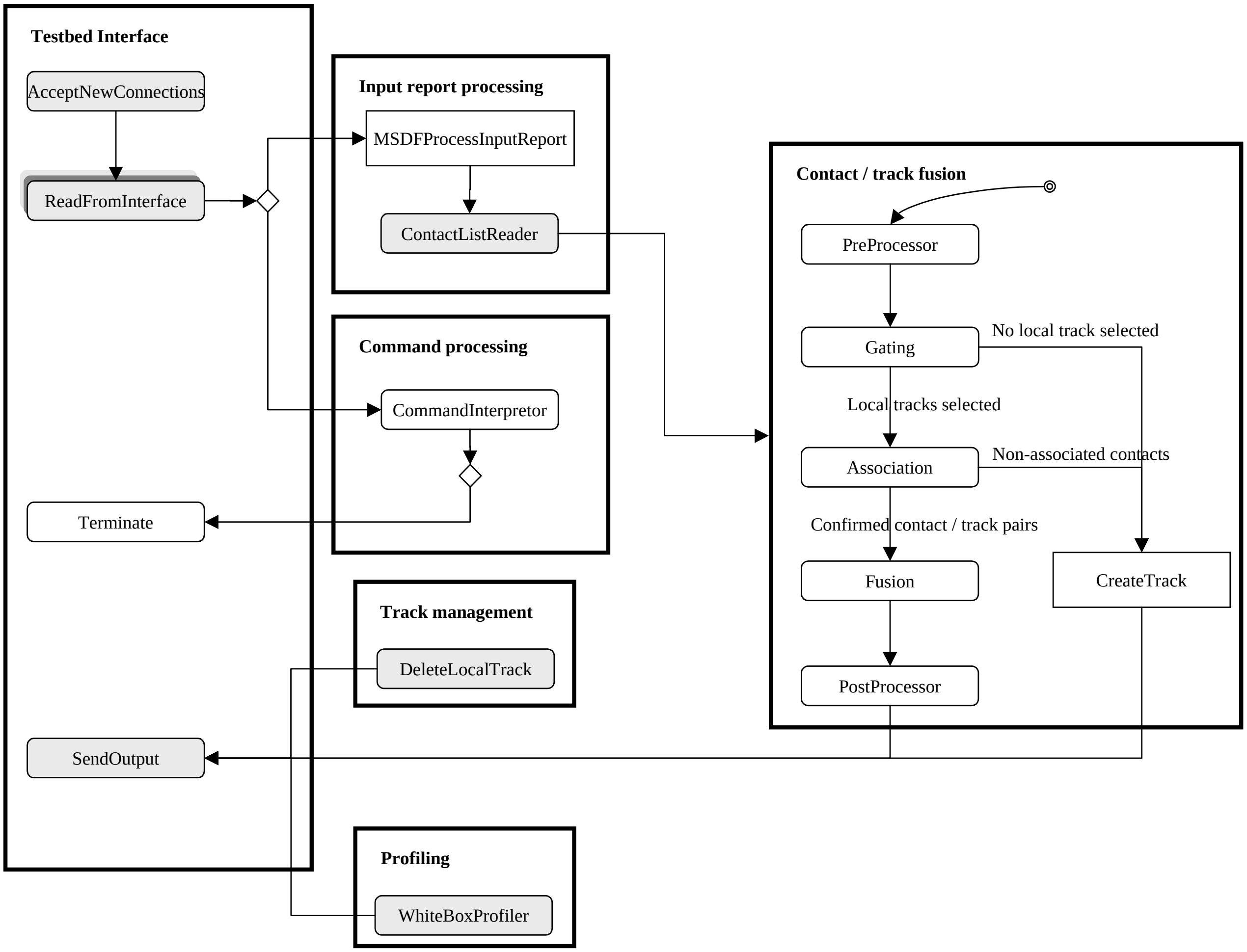



*Figure 16. Local MSDF data flow*

Whenever an input report comes in, it is passed along to the MSDFProcessInputReport function by the parallel agent ReadFromInterface, which is part of the TestbedInterface module. A ContactBuffer object is constructed from the message and put in a list on the blackboard. This list is read by the ContactListReader parallel agent. This agent encapsulates buffers into CONTACT_BUFFER data types and drops them on the blackboard for the PreProcessor agent to take and thus starts the data fusion process. However, the ContactListReader will instantiate new data only when the previous contact buffer has been completely processed. Its role is to make sure that no two buffers are processed at the same time.

The fusion process itself is divided up into three agents: Gating, Association and Fusion. Two other agents perform pre- and post-processing.

1. Pre-processor agent: The CONTACT_BUFFER data triggers the PreProcessor agent. This agent is primarily responsible for processing adaptive fusion commands sent by the user.

When finished, it puts on the blackboard a GATING data type. The track list is locked for the remainder of the fusion process.

2. Gating agent: This agent performs all gating computation prior to the association. First, it selects a sample of local tracks for candidate association with contacts. Then, it creates all possible contact/track pairs from the list of contacts and the selected local tracks. This information is maintained in the PAIR_LIST data type. Next, for each contact/track pair, a positional probability of association (chi-square statistical distance) and the Dempster-Shafer conflict are computed and combined into a total probability of non-association. Finally, an assignment matrix is built with probabilities of non-association and put on the current blackboard as the ASSIGNMENT data type. If no tracks were selected, tracks are created for each contact and the process stops there. The TRACK_LIST data type is activated so the PostProcessor agent can be triggered.

3. Association agent: This agent is triggered by instantiating the data ASSIGNMENT, which calls the function AssociatePairs of the appropriate associator. The function ProcessAssociations will eliminate the non-selected pairs from the pair list and create tracks for unassociated contacts. Finally, the data PAIR_LIST will be activated.

4. Fusion agent: Activated by the PAIR_LIST data type, this agent takes all confirmed pairs in the list and fuses the contact to the track by calling the appropriate tracker's function. Positional information is fused, followed by identity information. Positional fusion will occur only if the contact is in the future of the track or lags behind by a small amount of time. Every combination of XY and BO track is treated. That is, if a BO track is updated by an XY contact, a new XY track will be created and the old BO track will eventually be deleted. Vice versa for a BO contact updating an XY track. For identity fusion, improved consecutiveness is checked and a decision to fuse, or not, is taken based on the history of the track. For example, if the same information has been previously fused, no fusion will take place if the conflict is below some user-defined threshold. If the conflict is above the threshold, fusion will take place in the hope of recovering from a previously wrong association.

5. PostProcessor agent: This agent is primarily responsible for sending information from the LocalMSDF to other blackboards. It builds ReportedTrack objects and puts them onto the GlobalMSDF and InformationManagement blackboards. This agent also sends Black Box data messages to the RTD. When profiling is activated, it sends WhiteBox data. It also checks for missed deadlines. Finally, it is responsible for cleaning different data structures.

Track management in Local MSDF is done by the parallel agent DeleteLocalTrack, which periodically scans the list of local tracks and deletes those that have not been updated for a given time. Whenever a track is deleted, a message is sent to IM.

When white box profiling is requested by the user, the parallel agent WhiteBoxProfiler collects system information like the CPU load and memory utilization and sends this data to the RTD.

### 5.3.2 Global MSDF

In the following subsections, the agent-based implementation of Global MSDF is presented. First, the data flow between the various agents is presented in Figure 17, which provides an overview of the system by presenting how functional blocks are related to each other. Then, a detailed description for each functional block is presented. Finally, a description of the main data types of the Cortex implementation is presented together with the associated agents.

This page intentionally left blank.

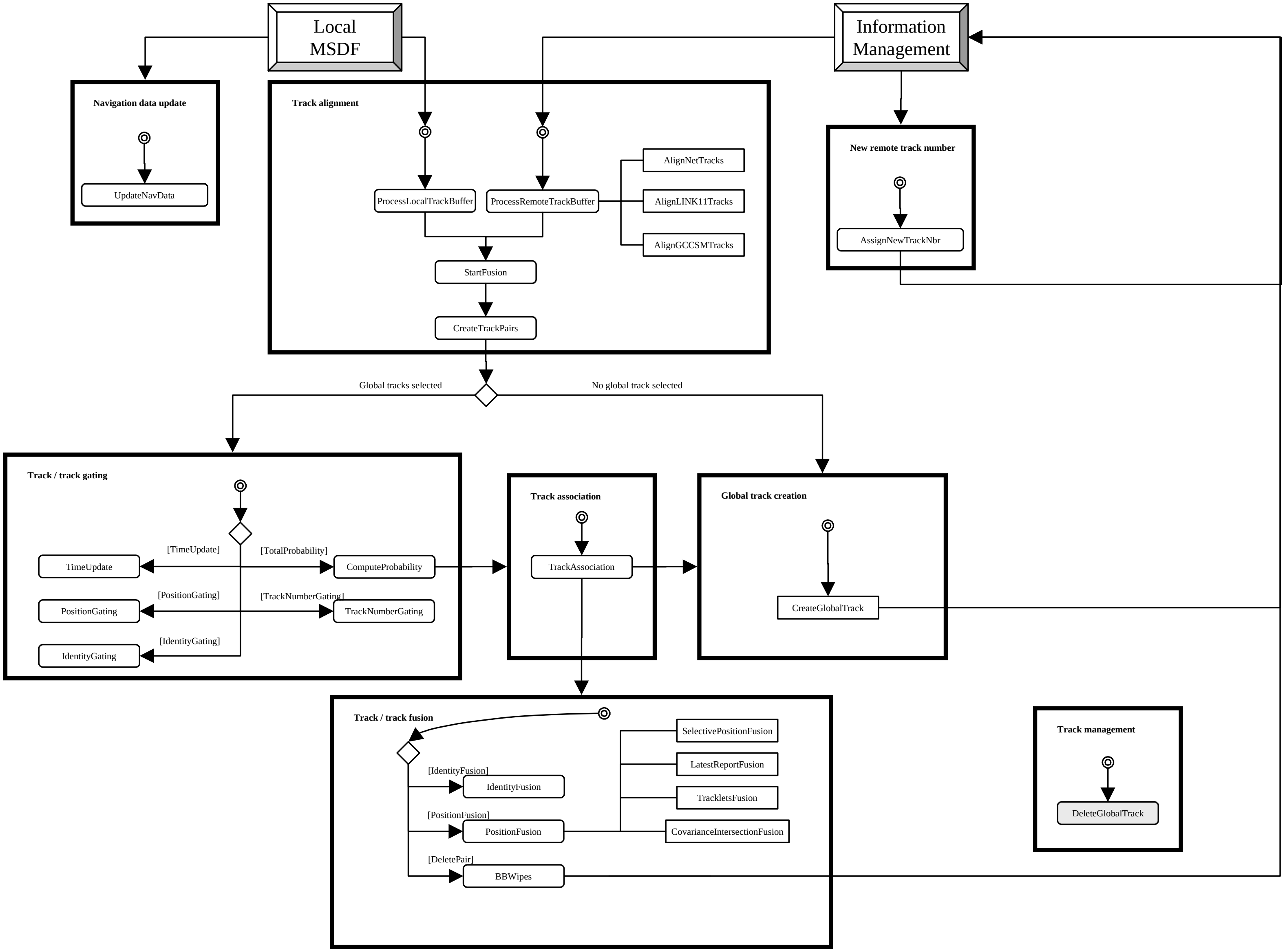



*Figure 17. Data flow diagram for Global MSDF*

Navigation data contains positional information on the current platform, i.e., the current position and velocity of the platform relative to the Data Link Reference Point (DLRP). This information is received by Local MSDF, which relays it to Global MSDF, where agent UpdateNavData processes it. Navigation information is used by Global MSDF whenever the position of a target relative to the ownship is needed.

Track alignment is performed for each reported track buffer received from either the Local MSDF or IM blackboards. The sequence of agents is as follows:

1. Agent ProcessLocalTrackBuffer processes buffers of reported tracks received from Local MSDF. Local MSDF sends its tracks in the correct format for reported tracks. The agent then takes the buffer and passes it to the next step. If configured by the user, this agent enforces hard association between local and global tracks.

2. Agent ProcessRemoteTrackBuffer processes buffers of reported tracks received from remote sources (Net, Link-11 or GCCS-M). Track information is contained in messages, which are first decoded and then processed.

3. Agent StartFusion then performs some processing prior to data association, e.g., processing Adaptive Fusion commands (changes in the association algorithms for some tracks).

4. Next, agent CreateTrackPairs selects a sample of candidate global tracks for association with the reported tracks. When no global tracks are selected in the region of the buffer of reported tracks, new global tracks are created. When global tracks are selected, the CreateTrackPairs agent puts all candidate track/track pairs on the blackboard. These pairs will then go through the track/track gating processes.

The track/track gating block computes probabilities of association and combines them into a total probability of non-association. Agents are executed in the following order:

1. Agent TimeUpdate: This agent performs a time update of the global track for each track/track pair in the list, using the user-selected tracker. When the reported track is in the future of the global track, the latest state vector and covariance matrix of the global track are time-updated. When the reported track is in the past, the history of the global track is scanned until the time difference between the reported track and the history point is positive. Then the history point is time-updated to the time of the reported track. In the case where the oldest history point is still in the future of the reported track, this history point is propagated backward in time.

2. Agent PositionGating: This agent performs position gating for each track/track pair in the list. Based on the pair type, it calls the appropriate function, which computes a statistical distance between the reported and global tracks. Then a probability of association is computed and stored in a member of the pair.

3. Agent IdentityGating: This agent performs identity gating on each track/track pair in the list. It takes the proposition set of the reported track and computes its Dempster-Shafer conflict with the proposition set of the global track. The conflict is stored in a member of the pair.

4. Agent TrackNumberGating: This agent performs track number gating for each track/track pair in the list. The computation is performed by the TrackNumberMgt object of the global track of the pair, and is stored in a member of the pair.

5. Agent ComputeProbability: This agent combines the positional probability of association, the Dempster-Shafer conflict, and the track number weighting factor into a total probability of non-association. It creates an assignment matrix, stores the computed probabilities of non-association and puts the matrix on the blackboard.

The track association block is composed of only one agent: TrackAssociation. This agent resolves the assignment matrix and assigns reported tracks to global tracks. It uses the associator attached to the TRACK_TO_TRACK_ASSOCIATION data type. The associators themselves are global variables (G_NNAssociator_ptr, G_JVCAssociator_ptr and G_MHAssociator_ptr). When a reported track is assigned to a global track, the associated track/track pair is confirmed. New global tracks are created when requested by the associator, e.g., when a reported track is not associated with any global track. Unconfirmed pairs are deleted. For each confirmed track/track pair, the associated local or remote track number is modified in the global track, if necessary. This information will be processed after fusion is completed.

Confirmed track/track pairs are processed by the track/track fusion block, where both position and identity fusion are performed. The following agents belong to this block:

1. Agent IdentityFusion performs the truncated Dempster-Shafer fusion. This agent performs identity fusion for each track/track pair in the list. This agent takes the proposition set of the reported track and fuses it with the current proposition set of the global track. Each global track has an instance of class GlobalIdentity_c, which manages how the reported identity information is fused into the identity of the global track. This agent also adds the reported track number to the track's history.

2. Agent PositionFusion: This agent updates the position of global tracks with the user-selected position fusion algorithm. Currently supported algorithms are Latest Report Fusion, Selective Position Fusion, inverse Information Filter, and Tracklets Fusion. In the future, this agent should also support Covariance Intersection. For each fusion algorithm, a function is called, and it is this function that actually performs the fusion.

3. Agent BBWipes: This agent performs post-processing. It is executed once the processing of the list of pairs is completed, i.e., after creation or fusion of tracks. It also removes the instance of data type REPORTED_TRACKS_BUFFER, which should be unique to the blackboard. If BO tracks were updated and there

are other BO tracks in the global track list, a new instance of REPORTED_TRACKS_BUFFER containing BIF tracks is created and activated on the blackboard. On the other hand, when no BIF track is created, a new REPORTED_TRACKS_BUFFER is extracted from the buffer list REPORTED_TRACKS_BUFFER_LIST and put on the blackboard. Then, if an instance of data type TRACK_ASSOCIATIONS exists on the current blackboard, it is extracted and put on the IM blackboard, if present. When the MH associator is used, it asks which tracks are to be presented to the user. Finally, Black Box data for the updated tracks is sent.

A new global track is created when agent CreateTrackPairs does not select a global track or a reported track is not assigned to an existing global track, according to agent TrackAssociation. For this purpose, these agents call function CreateGlobalTrack, which instantiates global tracks with the appropriate attributes and members. The new tracks are put on the IM blackboard.

Track management is used to delete global tracks that have not been updated for a given period of time. Only one agent belongs to this block, DeleteGlobalTrack, which is a parallel agent. This agent scans the list of global tracks and deletes those that have not been updated for a given time. This agent is intended to run in parallel, since it works with an infinite loop that periodically wakes up and scans the list. The global variable G_track_deletion_period, which is a user-defined parameter, sets the period of the scans. The deletion time difference depends on the type (BO or XY) and the category (air, surface, sub-surface) of the track. All these time difference parameters are user-defined. Each time a global track is deleted, a global track deletion message is sent to the RTD and the IM blackboard.

Global MSDF is notified when IM assigns a new remote track number to a local track to broadcast. Agent AssignNewTrackNbr assigns a new remote track number to a currently existing global track. IM should put the data type that triggers this agent when it broadcasts a local track not assigned to any remote tracks. This agent takes the information provided and updates the association maintained in the list of global tracks. In turn, it confirms the new association to IM by putting TRACK_ASSOCIATIONS data on its blackboard.

### 5.3.3 Information management

In the context of the test-bed, IM is a rules-based module that manages information transfers from and to the Net Manager and Communication Manager. IM performs the following tasks:

1. Processes requests for data coming from the Net Manager
2. Periodically sends track data to the Communication Manager and receives track data from it
3. Filters the input/output of data according to activated filters
4. Passes information received on remote tracks to the Global MSDF blackboard
5. Determines $R^2$ based on local and remote track qualities

6. Manages the message prioritization process.

In doing so, it must work with Local MSDF, Global MSDF and the network (i.e., the Net Manager and Communication Manager).

IM must work in two modes. First, in the $R^2$ mode it reports only local tracks for which it has $R^2$. In the second mode, it reports all its local tracks or global tracks irrespective of $R^2$.

For TQ comparison purposes, IM must maintain a list of associations between local, global and remote tracks. The association is performed by Global MSDF, not IM, and is therefore received from Global MSDF as soon as this information is ready. De-associations are taken into account by the same mechanism.

When polled by the Net Manager or when it needs to send information to the Communication Manager, the IM must scan its local or global list and decide whether a particular track should be sent to the appropriate application. The decision is based on different filters applied sequentially. The filters can be applied on the $R^2$ and the association status. When in $R^2$ mode, the IM will send all tracks for which it already has $R^2$ and an association status other than unknown. For all other tracks, checks are made to see whether it should take over $R^2$. These checks are described further below. When the report all mode is selected, all local tracks are sent.

When a newly created local or global track is received, IM must wait until Global MSDF reports on its association status. If the local track is associated with an existing remote track, IM should compare track qualities in order to determine whether the current platform should take over $R^2$. If the track is not associated with any remote tracks, IM assigns a new remote (link) track number and sends it to the Net Manager or the Communication Manager. IM must monitor the TQ of reported local tracks and global tracks in order to take over $R^2$ whenever necessary. IM must also maintain up-to-date information on platform navigation data, as this information is also sent to other PUs at the beginning of every transmission with a NetPUData message. This information will be periodically updated from Local MSDF.

The agent MessageReader does the network interface. The network message processing part is responsible for decoding all messages coming from the Net Manager or Communication Manager and for taking appropriate action. The SendTrack function is called to make the decision as to whether to send a track based on $R^2$ and for all other outgoing communication. The other agents, UpdateTrackPairs, UpdateTrackList and UpdateGlobalTrackList, are responsible for decoding data coming from the Local MSDF and GlobalMSDF blackboards.

# 6. Information prioritization

In real time systems, processing time is one of the most important constraints. If there is too much processing to do in the amount of time available, rules must be defined to determine what must be done first and what can wait or be put off to an unspecified time.

Message prioritization makes the system process the most relevant information first. The mission context of the platform and the content of the information determine the relevance of information. The rules used for prioritization are based on the Canadianization of Handbook 5 for Maritime Information Management. Hence, platforms may have different tactical pictures depending on these priorities.

Since the current test-bed focuses on level 1 data fusion, much of the information that can be exchanged between collaborative platforms is not of interest. In fact, a level 1 data fusion system is interested in track information like position and identity. However, other types of information are relevant to higher levels of data fusion systems. Such information may be meteorological observations, geographical data, operation orders, force planning, tactical images and tactical maps (air traffic corridors, commercial air routes, hazard zones, platform status, and engagement status).

## 6.1 Different information types exchanged

The test-bed supports the following items from Table 1 of the Canadianization of Handbook 5, paragraph 3.1:

a. 1A, 1B, 1C: Information about an AIR track

b. 2A, 2B, 2C: Information about a SURFACE track

c. 4: Information about the ownship

The information on the first two items must always contain all three information types: Kinematics (A), Identity (B) and Amplification data (C). Currently, the test-bed transfers all this information between platforms simultaneously.

To test the prioritization system, only three types of information are used to compute the message priority: AIR, SURFACE and OWNSHIP. An alternative is to compute the priority for each subtype (A, B or C) and associate the largest priority. This way it is possible to distinguish seven types of information, hence seven levels of priority.

Since the test-bed focuses on level 1 data fusion, only three different items out of the 53 types of messages identified in Table 1 of the Canadianization of Handbook 5 are pertinent to make available to platforms.

## 6.2 Mission context

All mission goals described in the Canadianization of Handbook 5, paragraph 3.2, are implemented. The design allows the number of goals to be easily increased. There is a list of goals and, for each, a database file with information about it. The content of the database file

can be changed without having to modify the code. To efficiently test the three different types of messages, a new goal was created to favour Surface tracks over Air tracks.

Other information is also used to compute the priority of incoming messages, like stable and dynamic conditions.

a. Stable conditions: Constant or very slowly changing circumstances describing the conditions or states of a PU.

   1) War/Peace: A PU is either in a war or peace condition. Mutually exclusive condition.

   2) Littoral Water/Pilotage Water/Open Sea: This is the area of operation of a PU. Mutually exclusive condition.

b. Dynamic conditions: These are changing or unpredictable circumstances describing the conditions or states of a target, as estimated by a PU.

   1) Close Point of Approach (CPA)

   2) Bearing/homing.

These conditions belong to a higher level of data fusion and are outside the scope of the current project, and are therefore not implemented. It will be more natural to handle them at the STA/RM level rather than at the MSDF level.

## 6.3 Context assessment database

All tables in the Canadianization of Handbook 5, paragraph 3.3, are used as databases depending on the chosen goal. There is one more database developed for a goal that favours Surface tracks instead of Air tracks. These databases are loaded at system initialization.

## 6.4 Information management filters

The test-bed supports filters in the IM module, as proposed by the Canadianization of Handbook 5, paragraph 5.1.4.2. These filters are used to keep only relevant information according to the mission context of the PU. The filters can be applied to either the IM output or the IM input, to select what is sent or what is received, respectively.

Currently, the test-bed supports identity filters based on a reported track's allegiance. Future investigations may use filters based on target range, track quality, etc.

## 6.5 Computing priority

The IM module uses Equation 6.1-1 from paragraph 6.1 of the Canadianization of Handbook 5 to compute the priority of incoming information about each track or PU. IM gets the following parameters, depending on the received information type, from the mission context

goal, the war condition, water area, and the dynamic condition of its own PU. These values and their associated weights are used in the following priority equation:

$$priority(i,\alpha) = wI_{\alpha} \cdot (w_P P_{i\alpha} + w_Q Q_{i\alpha} + w_T T_{i\alpha})$$

where:

$w =$ weighting factor for priority

$w_p =$ weighting factor for Potential

$w_Q =$ weighting factor for Quality

$w_T =$ weighting factor for Timeliness

$I_{\alpha} =$ importance of context $\alpha$

$P_{i\alpha} =$ potential of information item $i$ in context $\alpha$

$Q_{i\alpha} =$ quality of information item $i$ in context $\alpha$

$T_{i\alpha} =$ timeliness of information item $i$ in context $\alpha$

In the current implementation all weighting factors are set to one.

## 6.6 Communication between PUs

### 6.6.1 Datalink communications

The NetManager application manages data transmission between platforms in a Link-11 like manner. Each platform is polled periodically to send information from all its tracks to other platforms (Figure 18).

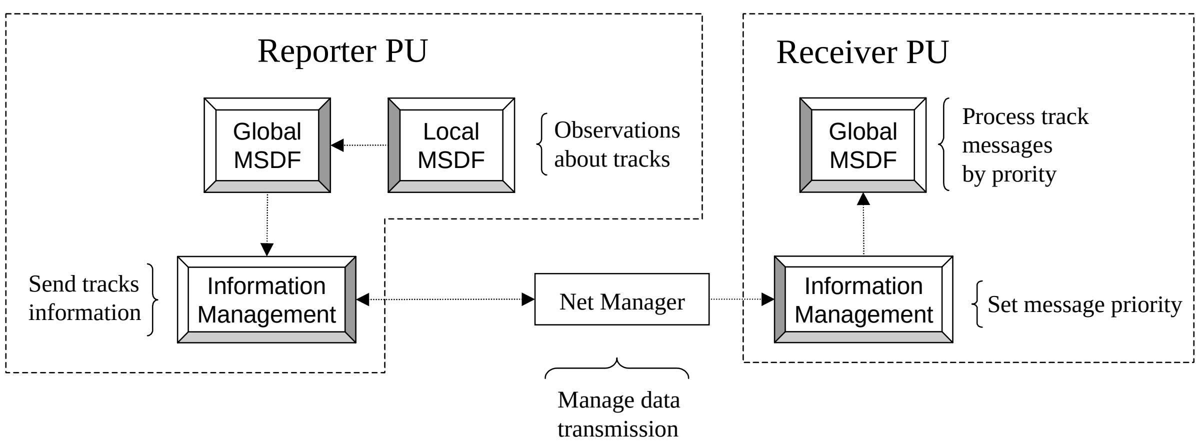



*Figure 18. Data exchange between PUs*

The subsets of tracks are sent to Global MSDF one by one, with the computed priority value. Global MSDF receives and stores the buffer in a container structure that can queue and restore buffers using various scheduling algorithms based on priority (see Figure 19).

Currently, only a very simple scheduling algorithm is implemented. All buffers queued in the highest priority list must be processed before buffers in lower priority lists can be processed. This ensures that highest priority buffers are always processed first. On the other hand, lower priority buffers may never be processed and keep accumulating because higher priority buffers keep coming in at a higher rate than they can be processed.

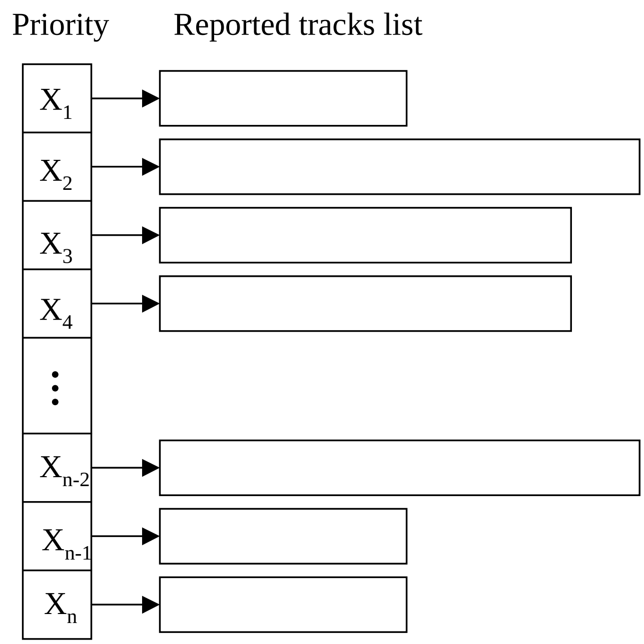



*Figure 19. Representation of track buffers stored according to priorities*

Other scheduling algorithms may be investigated in the future.

### 6.6.2 Point-to-point communications

In the context of NCW, a PU may want to send information to one and only one PU, e.g., some track information or an engagement command that is only of interest to the designated PU. Moreover, such communication may be unidirectional, i.e., a PU sends information to another PU and the latter may not have the capability or need to send information back. In these cases, point-to-point communications are required between PUs of the task force.

The test-bed was enhanced to provide point-to-point communications between PUs.

The approach taken allows many variations on the network topology, i.e., which PUs may send information to given PUs. The network topology must also support the capability to communicate with both unidirectional and bidirectional links between PUs. Finally, it is required to mix link-like communications with point-to-point communications. An example of a network topology is given in Figure 20.

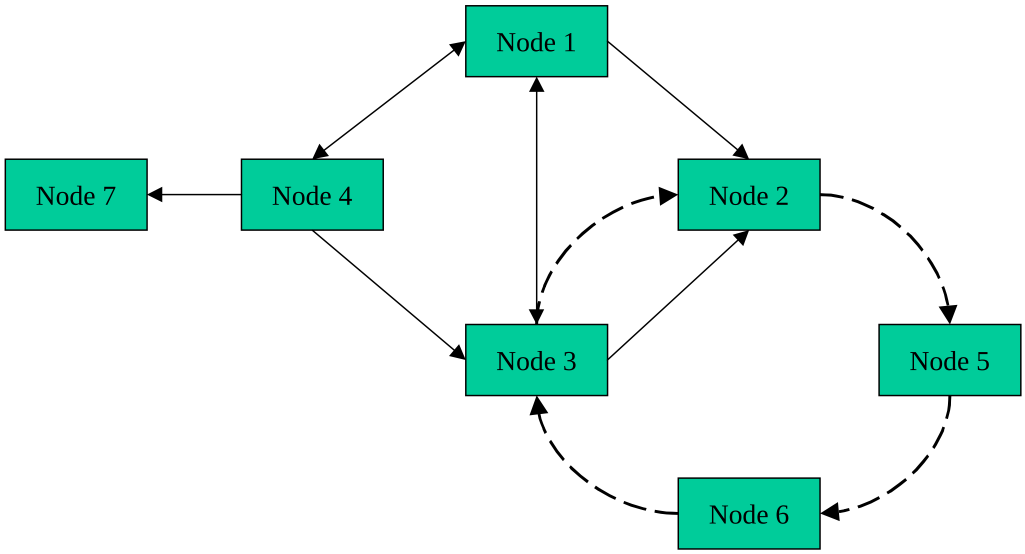



*Figure 20. Example of a network topology*

In this example, nodes 2, 3, 5 and 6 are connected to a link-like network (broken lines), while nodes 1, 4 and 7 are not. On the other hand, nodes 1, 2, 3, 4 and 7 have point-to-point communications (straight lines) between them. As the arrows indicate, node 1 can send information to nodes 2, 3 and 4, but node 2 sends no information through point-to-point communication links. Similarly, node 4 receives and sends information from and to node 1, but does not receive information from node 3, although it can send information to it.

Currently the test-bed uses CASE-ATTI for target generation and sensor simulation. CASE-ATTI has the concept of platforms with sensors. In the test-bed, these platforms are the fusion nodes on which CCIS applications participate. For example, platform 1 may have a Local MSDF module, a Global MSDF module and a STA/RM module, while platform 2 may only have a Local MSDF module and a Global MSDF module. The concept of fusion nodes should not be limited to platforms with sensors, that is, a fusion node may not have sensors attached to it. These nodes are not modelled by CASE-ATTI, but should be available during a given simulation. These fusion nodes do not have sensors, but will receive information from other nodes, either via the link-like network or through point-to-point communications.

When the Net Manager and multi-platform concept was introduced to the test-bed, only local track data was broadcast on the link-like network. With the introduction of point-to-point communications, the system should also have the ability to transfer global track data. This is required when information is relayed via one node before arriving at another node. In the example of Figure 20, global track data must be passed from node 4 to node 7 in order for node 7 to receive track data on tracks that were seen only by node 1 sensors. However, since the node 4 sensors may not see targets seen by node 1, if local track data is passed between node 4 and 7, node 7 will have no information about these targets.

The Net Manager takes care of link-like communications, as described previously. For point-to-point communications, a new application was developed: the Communication Manager. Only one instance of this application exists during a given simulation and maintains connections to the various instances of the CCIS applications that were connected to it upon

system start-up. This application must know the network topology in order to be able to route information received to the appropriate recipient. The Communication Manager is a specialized router for node-to-node information.

In the test-bed, the network topology is configured using a network topology file. This file should list all platforms of the simulation, i.e., fusion nodes with sensors as modelled by CASE-ATTI. It could also contain nodes with no sensors. This network topology file is provided to the Communication Manager that uses this information to build its routing table. The topology file associated with the example of Figure 20 is as follows:

```
1    Node1      no    2 3 4
2    Node2      yes
3    Node3      yes   1 2
4    Node4      no    1 3
5    Node5      yes
6    Node6      yes
7    Node7      no
```

Each line configures a given node. The first column contains the node identification number. The second column contains the node name. The third column tells whether or not (yes or no) the node participates in the link-like network. Finally, the last column lists all nodes to which information from the given PU should be passed. Note that these connections are not bidirectional. To have bidirectional information exchange, both PUs must list their peers in their list of point-to-point connections. For this purpose, in the example above, node 1 has node 4 in its list and node 4 has node 1 in its list. Finally, node 7 is not part of the CASE-ATTI simulation and therefore has no sensor attached to it. It could, for example, be seen as a command centre collecting information from various PUs.

In the test-bed, the Communication Manager is a unique application to which CCIS applications are connected. Upon system start-up, the Communication Manager is started with the network topology file and it waits for connections from CCIS applications. Among the CCIS applications, it is IM that is responsible for connecting to the Communication Manager at start-up.

Having point-to-point communications brings new challenges regarding data incest, especially when exchanging global track data. For example, let us assume two platforms exchange global track data through point-to-point communications.

At the beginning, platform 1 sends its global tracks to platform 2. The receiving platform fuses the information into its global tactical picture, which already contains local track data. That global tactical picture then contains the local track data from platform 2 and the global track data from platform 1. Platform 2 then sends it global track data to platform 1. On platform 1, this global track data is fused with the existing global tactical picture. However, the received track data contains information that is already fused in the global tactical picture. Fusing it without further processing causes data incest to appear in the system.

Regarding position fusion, Selective Position Fusion and Tracklet Fusion are two approaches that can be used to handle data incest problems by removing or remedying cross-correlation from the received track data. However, more investigations should be performed on this topic.

For identity cross-correlation, an approach similar to the Selective Identity Fusion should be investigated. But Selective Identity Fusion was designed to remedy local track data, so further investigation is required to modify the algorithm to remedy global track data.

# 7. Simulation environments

## 7.1 Evaluation of simulation environments

This section reports on a trade-off analysis for the most appropriate simulation environment (Modular Semi-Automated Forces (ModSAF), CASE-ATTI, HLA-based simulation systems, etc.) to support a multi-platform decision support system. The requirements for a variety of communication mechanisms and for closing the loop between the CCSs and their environments (i.e., CCS actions are reflected in the scenario, environment, own-platform, etc.) must be considered in this analysis.

A survey of simulation environments used within other distributed data fusion /decision support development environments found in the open literature was performed, keeping in mind the following questions about simulation environments:

- What are we trying to simulate and in what context?
- Can this simulation environment be integrated with the test-bed described in this report?

The main objective is to build a test-bed for developing, testing and benchmarking methods, techniques, algorithms, rule-based communication protocols and infrastructure to establish an MTP where the MTP is the combination of the LAP seen by a task force unit using it own sensors, the picture compiled through the sharing of information between the task force units (linked by a real/near real-time TDL, and the WAP) using information not controlled by the task force provided by HF, UHF radios or satellite. The various components of this test-bed should include a scenario generator, sensor simulators, non-task force simulators (e.g., GCCS), fusion capability and communication data exchange applications supporting multiple collaborating multi-sensor CCSs. Ideally these test-bed components should be designed in such a way that they are capable of linking with other Modeling and Simulation (M&S) applications. The selection of simulation environments that will support multi-platform development must also take into consideration the following capabilities:

1. Real-time capabilities, if required
2. Synchronization of common battlefield environment for the various sensors and non-task force simulators
3. Possibility to modify planned scenarios on the fly by introducing feedback mechanisms to support target manoeuvres, thereby enabling a missile launch, to perform avoidance manoeuvres, etc.
4. Modular M&S applications with well-defined specific capabilities that can facilitate simulation interoperability and reuse across a broad range of applications
5. Support Cooperative Engagement Capability (CEC) data exchange.

The DRDC Valcartier CASE-ATTI test-bed's scenario (target) generator and sensor modules have been used as a simulator for all the research performed and reported in the 3 DRDC-V reports on airborne maritime surveillance by the CP-140 aircraft (Documents TM-2004-281, TR-2004-282 and TR-2004-283) and also in STA/RM research. The CASE-ATTI test-bed was not designed to support real-time execution of scenarios, and its target generator and sensor applications are closely coupled.

The ModSAF simulation system provides the capability to create and control entities within a simulated battlefield. ModSAF was a joint effort between the U.S. Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Simulation, Training and Instrumentation Command (STRICOM). DARPA was responsible for architecture development, and STRICOM was responsible for enhancement, documentation and fielding to the simulation community. The first release of ModSAF to the simulation community was in January 1994 with a follow-on release of ModSAF 1.2 (including full DARPA Simulator Networking (SIMNET) functionality) in July 1994. Since that time ModSAF has gone through numerous releases, with its final major release of ModSAF 5.0 in 1998.

The ModSAF simulation system is currently being used on the Soar Intelligent FORces (Soar-IFOR) project. The Soar-IFOR project is investigating the application of Soar, a general cognitive architecture for developing systems that exhibit intelligent behaviour, to the modeling of real-time agent behaviour. The ultimate intent behind this effort is to develop automated pilots whose behaviour in simulated battlefields is nearly indistinguishable from that of human pilots, and to go beyond this, to develop generic agents that are readily specializable for this and other domains. For Soar-IFOR, ModSAF simulates an extensive list of entities including fixed and rotary wing aircraft, ground vehicles etc. Simulated entities can behave autonomously and can interact with each other, as well as manned simulators, over a network supported by Distributed Interactive Simulation (DIS).

The HALIFAX Class Operations Room Team Trainer (ORTT) project used a variant of ModSAF called Export Computer Generated Forces (ExportCGF), where any component of ModSAF that could be classified as US Technical Data was removed consistent with *International Traffic in Arms Regulations* (ITAR) restrictions. As a result of this, the ExportCGF did not include any sensor models.

In January 1996, the Computer Generated Forces (CGF) Assessment Working Group completed an assessment of the various CGF models in the Army and briefed the Army leadership of the results. From this assessment, the Army's CGF investment strategy was determined. The main recommendation was that a flexible, composable, Semi-Automated Forces (SAF) architecture be developed that could integrate the best features of existing CGFs such as ModSAF and the Close Combat Tactical Trainer (CCTT) SAF. In May 1997, the Deputy Commanding General of the U.S. Army Training and Doctrine Command, approved the Mission Need Statement for the OneSAF model. OneSAF will be fielded in 2004; however, the OneSAF program's purpose is to provide an interim product to support the simulation community. There have been two developmental drops, OTB A and OTB B, released to selected user labs prior to the release of OTB v1.0 in January 2001. OTB v1.0 provides numerous improvements and enhancements over its original baseline. Some of the notable improvements include dynamically loadable modules, the ability to separate the map from editors (dual monitor capability), an improved terrain cache, run-time data collection,

scalability (multi-threading), and portable scenarios. In addition, enhancements in radar modelling (high resolution) and behaviour representation in the areas of ground manoeuvre, aviation, fire support and engineering have been incorporated. The final OneSAF will be a composable, next-generation CGF that can represent a full range of operations, systems and control processes from individual combatants and platforms to battalion level, with a variable level of fidelity that supports all M&S domains. It will accurately and effectively represent specific activities of ground warfare (engagement and manoeuvre), Command, Control, Communications, Computers, and Intelligence (C4I), combat support, and combat service support. It will also employ appropriate representations of the physical environment and its effect on simulated activities and behaviours. The OneSAF Operational Requirements document can be found at http://www.onesaf.org/public1saf.html. The first release of OneSAF initiated the retirement of ModSAF.

Scenario Toolkit And Generation Environment (STAGE) developed by Virtual Prototypes Inc. is a software tool used to build and animate synthetic environments in real time. The synthetic environment can be composed of moving or stationary entities such as aircraft, ground vehicles, ships, radar sites, missiles etc. DRDC Valcartier has been evolving their Situation Analysis concepts exploration and the development of an M&S facility, called SEATS (Simulation Environment for the Analysis of the Tactical Situation), which is a test-bed for distributed simulations using STAGE. This effort is still ongoing, specifically in the area of introducing more realistic sensor models through the integration of STAGE with the BAE Systems Ship Air Defence Model (SADM). SADM is a software tool designed to simulate the defence of a task group (or a single ship) against one or more attacking aircraft and cruise missiles. It simulates soft-kill, hard-kill, and the interactions between them.

Rather than select one of these simulation environments, the next section will study the migration of the current test-bed in order to demonstrate collaboration between multiple High Level Architecture (HLA) compliant fusion systems, while continuing to simulate the scenarios and sensors using CASE-ATTI.

## 7.2 Fusion test-bed in a HLA operational federation

The role of simulations in today's development processes is increasing from the early prototyping to the final stage of production, and even after. Modern simulation software components are complex systems. They often reside on different dispersed computer systems linked by a network with different protocols.

These components must interoperate seamlessly. They have to be realistic and computationally fast. Often the best component simulator is a legacy application that must be integrated into a new environment. This is a difficult task fraught with compromises. It is often easier to implement a new simulation of a system component than it is to modify an existing one. The best-of-breed component is then not used.

The HLA strives to solve the two major problems mentioned above, namely the interoperability and the reuse of different components. The main function of the HLA is to bring together these geographically dispersed components running on a variety of platforms and to provide advanced time and data management. The HLA also provides a common

standard to promote the reuse of legacy applications by specifying an interface that each PU must comply with, thus reducing time and costs associated with development efforts.

### 7.2.1 High-level architecture

According to the United States Defense Modeling and Simulation Office (DMSO):

> “The High Level Architecture is a general purpose architecture for simulation reuse and interoperability. The HLA was developed under the leadership of the DMSO to support reuse and interoperability across the large numbers of different types of simulations developed and maintained by the DoD.”

The HLA was approved as an open standard by the Institute of Electrical and Electronic Engineers (IEEE) in September 2000 (IEEE Standard 1516). The HLA provides a standard that will hopefully reduce the cost and development time of simulation systems and increase their capabilities.

Kuhl et al. (Kuhl, 1999) describes the HLA as "... the glue that allows you to combine computer simulations into a larger simulation." However, this glue is an architecture for M&S applications, and not a simulator or a modelling tool, nor is it a rapid application development system for simulations. It does not provide any data display capabilities or user interfaces. It is *not* an implementation; it provides a framework.

The standard comprises three elements:

1. An interface specification that describes the way that compliant simulations will interact during operation

2. An Object Model Template (OMT) specification that specifies the form in which simulation elements are described

3. A set of rules that describes the responsibilities of the simulations and the infrastructure.

Many vendors have implemented the interface specification into what is called the Runtime Infrastructure (RTI). The DMSO also has an RTI, which is free of charge. This piece of software provides different services to the simulator components.

### 7.2.2 Nomenclature

The most important terms are briefly defined here.

1. **Federates**: Individual simulator components that the HLA tries to integrate. Each application talks to the others via the RTI. Note that a federate may communicate with or even start other applications that are not HLA-compliant. The federate will then act as a gateway for those applications to the running federation.

2. **Federation**: The whole ensemble of federates integrated together.

3. **Federation Execution**: A session in which a federation is running.

4. **Federation Object Model (FOM)**: An OMT that describes all pertinent data shared between participating federates in the course of a federation execution.

5. **Simulation Object Model (SOM)**: An OMT that represents a detailed description of a federate's capabilities and architecture. The FOM is a subset of the collection of SOMs defined for the various federates.

6. **Interaction**: A message describing an event sent to all subscribing federates by a publishing federate whenever the event occurs.

7. **Update**: A message updating new values of an attribute sent to all subscribing federates by a publishing federate at the end of each time-step.

8. **Runtime Infrastructure (RTI)**: A software that implements the interface specification and runs the overall simulation.

9. **Object Model Template (OMT)**: A template defined by the HLA standard. It defines the form, type and structure of the data and is the template used by the FOM and the SOM. It defines two formats: a human-readable one and a machine-readable one called the Data Interchange Format (DIF). The former is in tabular format, and the latter is a text-formatted file (the Federation Execution Data (FED)) used by the RTI software. It represents the FOM.

Keeping in mind the short description provided above of the key elements of an HLA, these concepts will be expanded further below in an overview of the HLA.

### 7.2.3 HLA overview

As stated above, the HLA consists of three components:

1. **Federation Rules**: ten rules that ensure the proper interaction of federates during a federation execution.

2. **Interface Specification**: defines RTI services and identifies the call-back functions each federate must provide.

3. **OMT**: provides a common method for recording information.

The HLA makes use of an object-oriented variation on the publish-and-subscribe paradigm to manage information sharing between federates. Persistent data are stored within object attributes. A federate that wants to send this data must register with the RTI as a publisher. Any other federate that wants to receive this data must register as a subscriber. Whenever a federate updates its data, it calls the appropriate function of the RTI. The subscribing federate will then be notified of the new values.

Non-persistent data are distributed through interactions, which are messages sent on a one-time basis. Again, the publish-and-subscribe design pattern is used.

The FOM is an OMT that describes all data pertinent to a simulation execution, i.e., all data shared between federates. It includes an enumeration of all objects and interaction classes pertinent to the federation, along with a specification of the attributes or parameters characterizing these classes. The FOM (in DIF format, it is called a FED file) is read by the RTI at initialization time.

A SOM is specific to a given federate and describes its entire range of capabilities: its objects, attributes and interactions. The SOM provides an indication of the suitability of a simulation component for participating in a particular federation. Like the FOM, it is documented in accordance with the OMT.

The RTI is a piece of software that is not part of the HLA standard, but is rather an implementation of the HLA interface specification provided by different companies, whose purpose is to provide services to the federates.

One of the main tasks of the RTI is to manage communication between each and every federate. This communication is done through two classes, RTIAmbassador and FederateAmbassador, provided by the RTI software. The former is used by federates to invoke RTI services and the latter by the RTI to call back on the federates. The RTIAmbassador class contains a large set of public methods for all possible requests. The FederateAmbassador is an abstract class with virtual methods. Each federate must inherit from this base class and implement the virtual public member functions to serve as call-backs. Upon joining a federation, every federate must hand over an instance of a locally derived FederateAmbassador.

The six management service areas of the RTI are described in the following paragraphs.

1. Federation management: This service provides coordination of federation-wide activities throughout the life of a simulation: creation and destruction of a federation execution, federates joining and resigning from a federation and coordination of the synchronization points of a federation. Note that a federation execution process must exist before any federate can join.

2. Declaration management: This service handles the publication and subscription of objects and interactions. Federates use this service to specify the type of data they will send or receive. This must be consistent with the FOM.

3. Ownership management: Only the federate owning an object instance may update its attributes. This service facilitates dynamic transfer of ownership. The RTI arbitrates transactions so that ownership is held by only one federate at a time. It offers both "push" and "pull" transactions.

4. Object management: Functions provided by this service include registering and discovering objects, updating and reflecting object attributes, sending and receiving interactions, and deleting and removing objects.

5. Time management: This service is used to prevent the execution of events in the wrong order. It controls the advancement of each federate's logical time by deciding when it receives time-stamped events. The RTI can manage either continuous (time-step) or discrete (event-driven) simulators. Note that this service has no reference to wall clock time.

6. Data Distribution management: Federates use this service to filter the transmission and reception of irrelevant data. It allows the distribution conditions to be specified for the specific data they send or ask to receive.

### 7.2.4 Modelling and Simulation (M&S)

M&S provides virtual duplication of products and processes, and represents these products in operationally valid environments.

A model is a physical, mathematical or logical representation of a system, entity, phenomenon or process. Examples are a plastic replica of a car or a mathematical equation that predicts the probability of an event occurring.

A simulation is a method for implementing a model over time. It is a technique used for testing, analysis or training, where a model can represent real-world systems or concepts.

Thus M&S is the use of models, including emulators, prototypes, simulators, and stimulators, either statically or over time, to develop data as a basis for making managerial or technical decisions. The terms modelling and simulation are often used interchangeably.

### 7.2.5 Blueprints

This section presents the architecture of the test-bed with an HLA-compliant fusion engine.

The first task is to identify the federates. We want to define what the model (see above definition) will be, and what physical or conceptual entity the federate will represent. We can make a federate out of a very small unit (e.g., an algorithm) or an entire multi-process simulation. In both cases there are trade-offs to be made. The former scenario offers the greatest flexibility, but at a very high network overhead and great software complexity. The latter scenario is not flexible at all but is easier to implement in terms of HLA coding overhead. The decision of where to wrap a process into an HLA federate will depend on why the simulation is being carried out.

The perfect-world scenario for simulating multiple cooperating platforms would have many small federates for every important subsystem of a PU. One federate would simulate the moving platform, i.e., would generate a new position and publish it. Another federate would be responsible for simulating certain platform sensors. Another would simulate the fusion engine, the STA engine, the communication layer and so on. This would allow a PU to be assembled and tailored the way we like. For example, a link communication scheme could be implemented by the use of a LinkCommFederate. Another communication scheme could be examined simply by replacing the federate with another one.

What we intend to demonstrate is information sharing between platforms through the RTI, so what we will model here is a track. This object would be updated by the PU that owns it, i.e., the PU currently tracking it. An update to this object for example could be a new position as computed by the fusion engine. The fusion process would be wrapped in a federate that operates on the Track. When tracking is handed off from one PU to another, ownership of the track object would be passed on to the new PU.

For simplicity and as a first prototype, the Local MSDF, Global MSDF and IM would all be wrapped as a federate. Sensor input would come from CASE-ATTI in the usual way, in the form of messages from our messaging library through sockets.

The different federates in this scenario would be (see Figure 21):

1. One or many fusion federates: A PU's MSDF, which would include the Local MSDF, the Global MSDF and IM. Published data would be classes of type track, would have attributes like estimated position, estimated velocity, probable identity, etc., and would be time-constrained. Note also that a platform federate can spawn other applications, which are not necessarily HLA-compliant. These helper applications would not be aware of the federation. They would use their own protocol to communicate with a platform federate. This federate should communicate by way of a socket as is currently done with the CASE-ATTI simulator.

2. One or many communication network federates: responsible for emulating a particular network, be it Link-11 or any other. There should be one such federate for each platform. When appropriate, this federate would send an interaction with all necessary parameters that other platforms should be aware of. It would broadcast the right data to listening platforms.

3. A manager federate: responsible for the synchronization of the federation. It would also be the time-keeper federate responsible for issuing a time advance request, and would be time-regulating. These functionalities could be included in another federate like the network federate or the target generator, but maximum flexibility would be achieved by having a separate federate do this job.

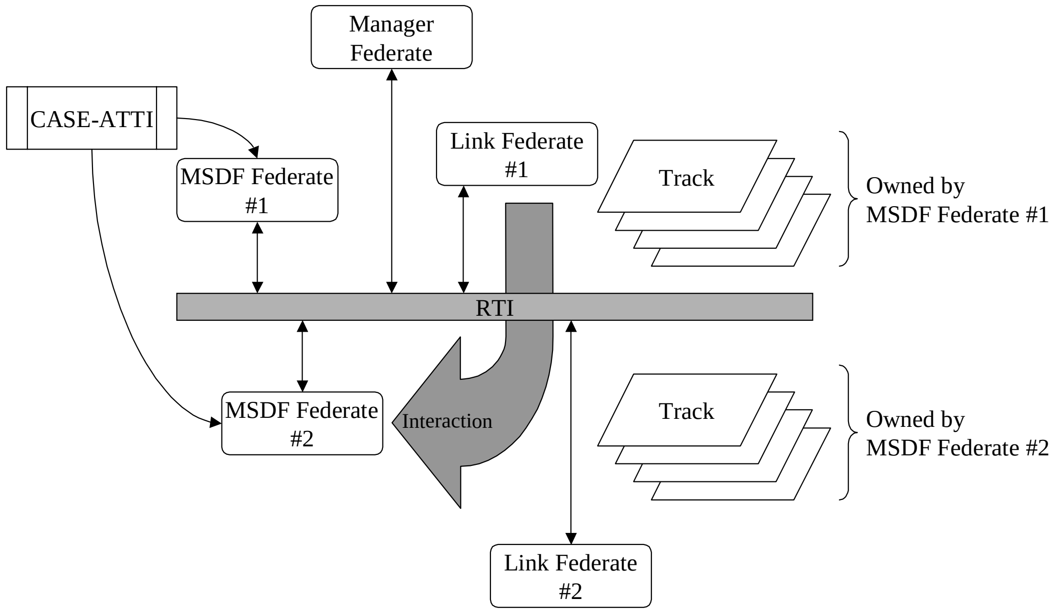



*Figure 21. Proposed high-level architecture*

Note that the SystemManager would not be a federate. It would simply be an application that brings up the federation, i.e., the different applications that are the federates. The RunTime Display (RTD), STIM Driver and CASE-ATTI would all use the same communication mechanism as they are using now. For reasons of simplicity, one should just make the inter-platform communication HLA-compliant and no more.

The modifications that would need to be made to the existing applications are now described.

As hinted above, the messaging library now in use would be obsolete for all communication between platforms. The communication federate would be responsible for publishing appropriate data to all subscribed platforms at the right time. Track classes are objects where all data produced by the Local MSDF are stored. They are in a repository accessible for anyone with access, but for now the only federate that would subscribe to it would be the communication federate. Eventually, any application that needs to analyze this data could do so.

Management services would be used to synchronize the federation. Specifically, synchronization points would be registered for key events in the federation's lifecycle (i.e., after initialization). These synchronization points are used as waiting points, where each federate, upon arriving at one, waits for the signal from the RTI saying that everyone has reached this point. This mechanism would replace the one used in the test-bed where the SystemManager waits for each application to send a SyncCommandMsg saying that it is ready to begin execution.

The major modification will undoubtedly be the implementation of the HLA interface. The abstract class FederateAmbassador must be derived from and implemented for each and every federate. Also, functions and methods to access the new data structure (i.e., track classes) would have to be written for each module/federate.

### 7.2.6 Current implementation

Since the HLA has very strict specifications that need to be implemented in order to function properly, and those specifications add a significant layer of complexity to the code, it was decided that only the MSDF part of the test-bed would be made HLA-compliant. Moreover, only the communication over the link-like network would be done the HLA way.

The adopted design is shown in Figure 22.

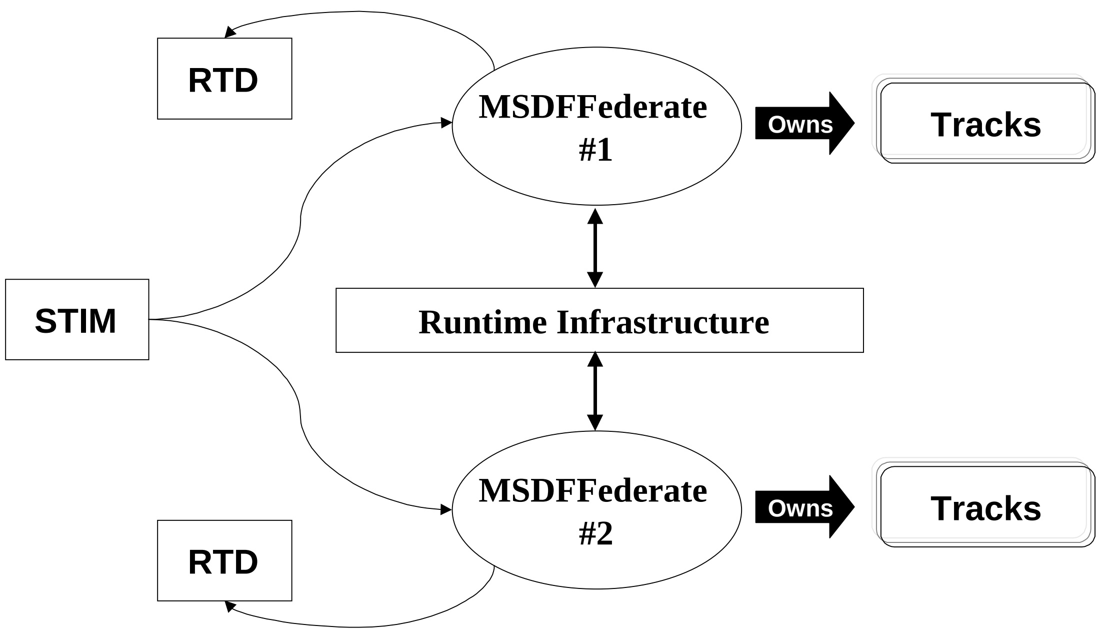



*Figure 22. HLA federation in the test-bed environment*

The MSDF, comprising the LocalMSDF, GlobalMSDF and IM blackboards, is wrapped into a federate that communicates with the Runtime Infrastructure (RTI). The RTI is responsible for bringing together all federates by mediating communications between them. In this case, the other federates in the HLA federation are other platforms MSDFs, or to be more precise, other HLA wrappers.

For simplicity, communication between all other participants and the MSDF is done as before, i.e., by messages from the test-bed Messaging library. This includes communications between the STIM Driver and the MSDFs, and between the RTDs and their MSDFs (as shown in Figure 22).

This design has an advantage that it is relatively simple to implement, because there is no need for any federates other than the ones mentioned. In other words, there is no need for an

HLA-compatible simulator. It is a proof of concept for how an MSDF box could be made HLA-compatible.

RTI 1.3 developed by the DMSO was used. RTI 1.3 is an implementation of the HLA specification that is free of charge but no longer supported. It provides among other things the RTIExec that is responsible for running the federation and relaying messages.

The wrapper written around MSDF is responsible for passing interactions between the IM and the RTI and for updating the HLA tracks (see Figure 23). These tracks are the repository of information that is updated by the MSDF and sent to other PUs when requested by the IM. They are HLA objects, in the distributed computing sense (see above), managed by the HLA Object Management service. They have attributes that contain all the information likely to be sent over the network. Among these attributes are positional and identity information.

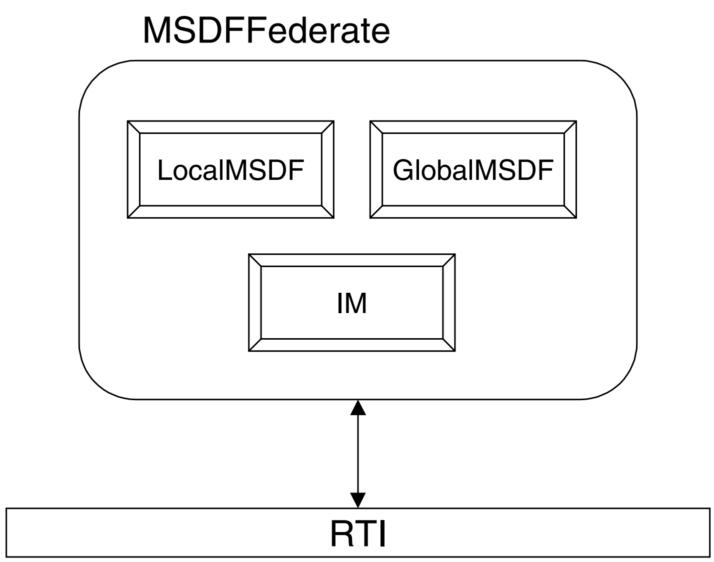



*Figure 23. The MSDFFederate wraps the LocalMSDF, GlobalMSDF and IM blackboards*

The transfer of data to other PUs is done when the sending PU updates a track object. The update is initiated by IM upon receipt of a Request Data message from the NetManager. This message is sent by the Messaging library and not via the RTI. When IM receives a request, it sends the MSDFFederate track information that needs to be updated. The MSDFFederate will then update the track objects by calling the appropriate function of the RTI. The RTI will notify all subscribers of the updated track object. Upon notification, the MSDFFederate that has subscribed to the updated object will send the track data to IM for further processing by the MSDF. The sequence diagram is shown in Figure 24.

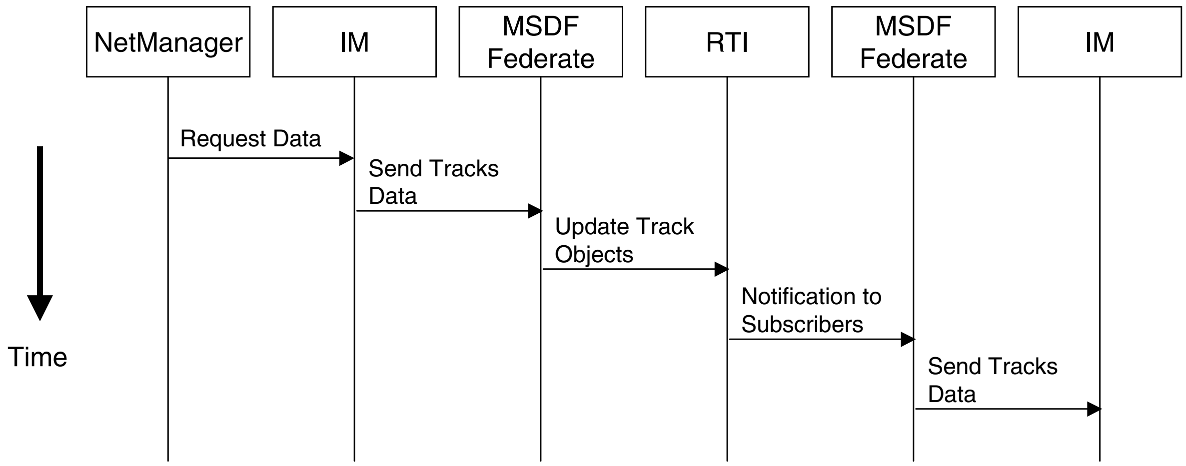



*Figure 24. Sequence diagram representing the interaction between different modules*

The RTI is also responsible for transferring other messages from the MSDFFederate. These other messages actually come from IM and are the NetDropTrack and the EndOfDataTransmission messages. The RTI also sends PU information such as its position and speed.

Ownership management is also taken care of by the MSDFFederate. One feature of HLA, which in this case was more of a drawback, is the fact that an object can only be updated by one federate at a time. In other words, only the owner of an object may update it and no one else. Consequently, care must be taken when $R^2$ changes. The rule for broadcasting information whenever the track quality is better than the currently reporting PU cannot be applied blindly because no two federates may update a track at the same time. Ownership has to be transferred first. Only when the PU with the best track quality has acquired ownership can it begin broadcasting.

Note again that this transfer of information to other PUs is the only one managed by the HLA. The STIM Driver still sends information via Transmission Control Protocol/Internet Protocol (TCP/IP) sockets using the Messaging library. Similarly, the MSDF sends information to the RTD via TCP/IP sockets using the Messaging library. These socket interfaces did not change. Thus, the current implementation of the HLA in the test-bed does not yet make the MSDF into a fully HLA-pluggable federate.

### 7.2.7 Conclusions

Migrating toward a completely HLA-compliant test-bed is not an easy task and would require a major rework of all the infrastructure code, i.e., communication and data structure. The HLA adds a considerable coding overhead to existing code but, if used properly, it holds the possibility for code reuse and interoperability.

The following are the conclusions from our analysis of lessons learned in the context of a federation comprising an MSDFFederate as part of a multi-federate platform interacting with HLA-compliant simulators and other federates:

a. The potential for code reuse is less than the potential for interoperability. That is, code developed for the HLA compliance part would not be reused outside of the MSDFFederate. Data fusion code is already well designed into classes and agents and its reusability is already proven without HLA. On the other hand, interoperability would be achieved by having strict interface definitions ensuring that all actors in the federation have the same view of the MSDFFederate.

b. The strict HLA rules for documenting object attributes ensure that all programmers understand each other and communicate in an efficient way. However, HLA cannot make "plug and play" a reality. To advertise it as such is to oversell it. In order to make systems compatible, negotiation of common data types and their meaning must still be done by a person. Specifically, a track object would still have to be known by all interested federates and the meaning of its attributes must be known by the programmer at implementation time. Changing even one of its attributes would cause a change in all applications using it.

c. HLA is probably not an appropriate architecture for simulations that just generate data for some single, external system or for simulations that cannot define their problem space as a collection of interacting objects.

Putting aside all these disadvantages, implementing an MSDF as a federate, if done properly and in concert with all other federate developers, would be advantageous for its interoperability alone. Namely, by defining a strict interface for MSDF box inputs and outputs, all participants would have to conform to it and would know how, because the HLA provides an architecture that is well defined and understood by all. That alone is not negligible.

# 8. Demonstrations of chosen algorithms

In two previous documents (TM-2004-281 and TR-2004-283), Local MSDF algorithms were described and their performance was demonstrated, while a third document (TR-2004-282) developed and benchmarked imagery classifiers.

In the same vein, this section will demonstrate Global MSDF algorithmic performance, and will report on enhanced imaging capabilities that have occurred since the above documents were issued.

## 8.1 Comparison of global track fusion algorithms

This section compares the track fusion algorithms described in sections 2.1.2 and 2.2.1, namely the inverse Kalman Filter, the inverse Information Filter, the Selective Position Fusion with TQ method and the Selective Position Fusion with covariance matrix method. For each of these algorithms, the EAKF is used to fuse either the reported track or the tracklet with the global track.

The data used for this comparison comes from a scenario that has two reporting units (see Figure 25). In the first test, only PU 1 broadcasts its local track data, while in the second test both PUs are broadcasting. All fusion methods are tested with broadcast data.

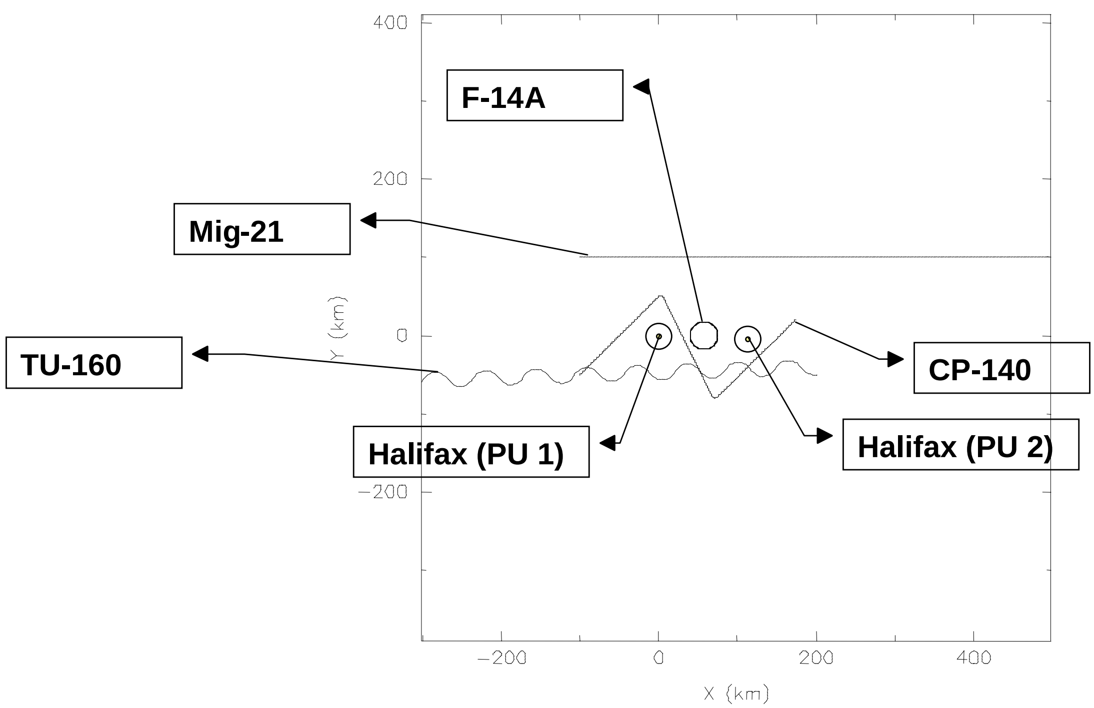



*Figure 25. Scenario description for position fusion analysis*

The analysis presented here focuses on two tracks of the scenario: the TU-160 target following an oscillatory manoeuvring pattern and the Mig-21 going in a straight line. The

polling cycle time of the net manager is 20 seconds. Each PU has an SG-150 sensor with standard deviations in range and bearing of $\sigma_r$ = 90 m and $\sigma_b$ = 0.06 rad, respectively.

The comparison will be based on the position track state and the trace of the positional part of the covariance matrix. The trace is the sum of the semi-major axis and the semi-minor axis of the positional uncertainty, and it is used as a measure of positional error.

### 8.1.1 Test 1: one PU reporting

This scenario has only one reporting unit, PU 1. Therefore, the optimal result for the global node must be the local track state and covariance matrix, since that is the only data available. The purpose of this test is to compare the various fusion approaches for consistency.

The output of each tracking method is compared with the Local MSDF tracking of the reporting unit. The position track state and the trace of covariance matrix are analyzed. In this scenario there is no need to look at the results of the SPF method (with covariance matrix), since they are identical to the local fusion node data. By looking at the Mig-21 going in a straight line, Figure 26 shows the difference in the covariance matrix traces.

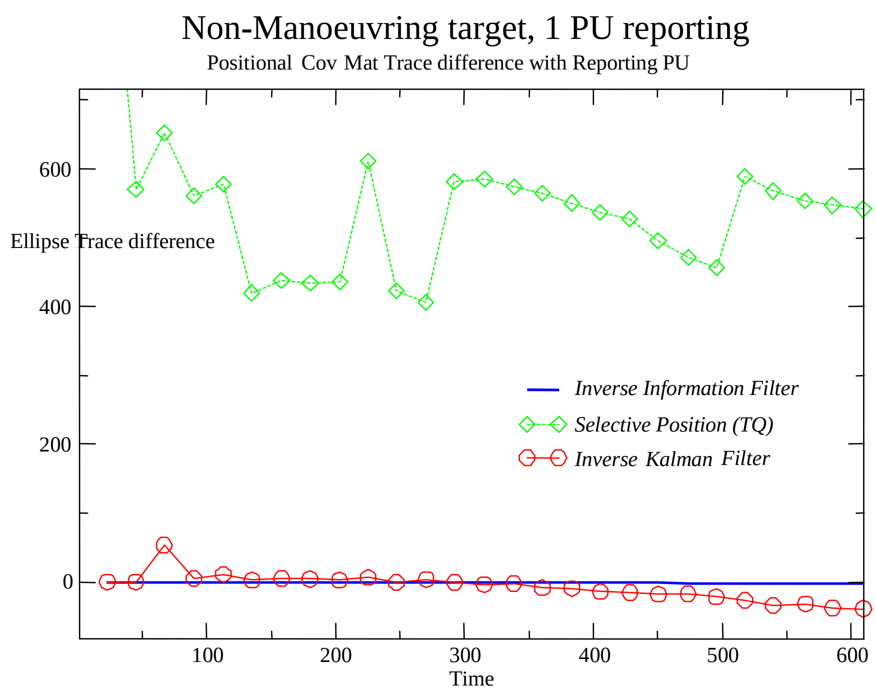



*Figure 26. Covariance matrix for non-manoeuvring target with 1 PU reporting*

The best methods are those that produce a result close to zero, where the error is like the local track error. The inverse Information Filter outperformed the two other methods and corresponds to local data, while the inverse Kalman Filter is close to the local data but not as

close as the inverse Information Filter. This difference is caused by process noise, which is assumed null by the inverse Kalman Filter. For the SPF with TQ, using TQ to generate contact-like covariance matrices causes the system to overestimate the fused covariance matrix. As seen in Figure 26, the covariance matrix trace for the SPF with TQ is greater than with the other algorithms, as expected.

Figure 27 shows the difference in the Y component of the state vector between the track fusion algorithm and the output of the reporting PU. The best methods are those close to zero where the position is like the local track. All the algorithms performed well and were very close to the local fusion node. The difference is less than 20 metres for inverse Information and Selective Position, while the inverse Kalman Filter oscillates more, up to 60 metres. Note that the target is located 224 kilometres from the farthest reporting PU.

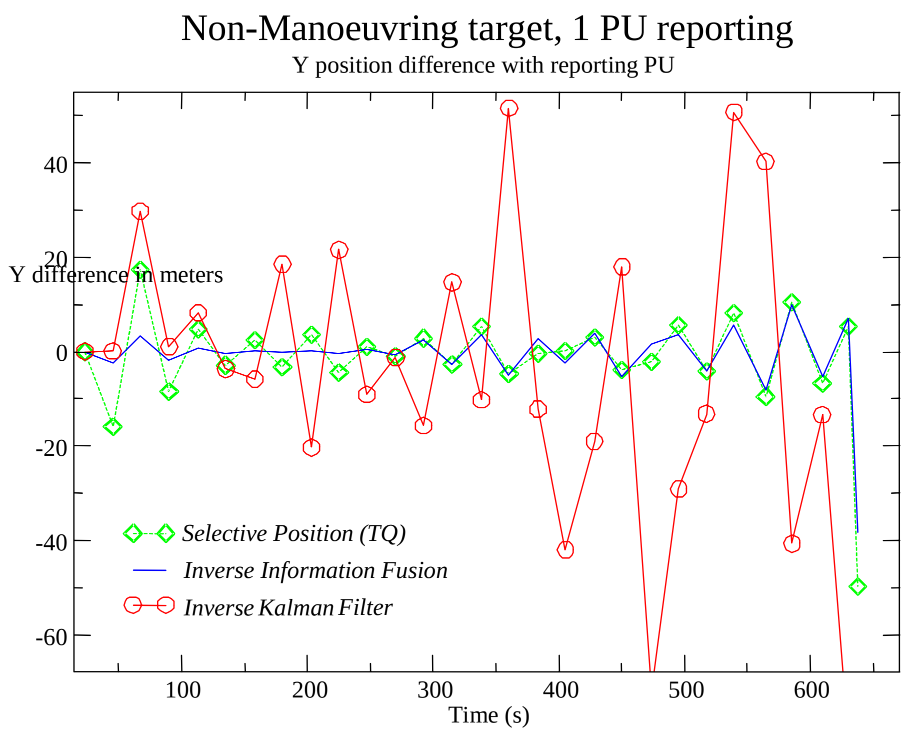



*Figure 27. Position for non-manoeuvring target with 1 PU reporting*

By looking at the TU-160 following an oscillatory manoeuvring pattern, Figure 28 and Figure 29 show the same behaviour as for the Mig-21. The inverse Information Filter is the best filter and follows the local fusion node. The inverse Kalman Filter is less accurate and shows more oscillations in position. The Selective Position is the worst for the covariance matrix, because it has a larger uncertainty.

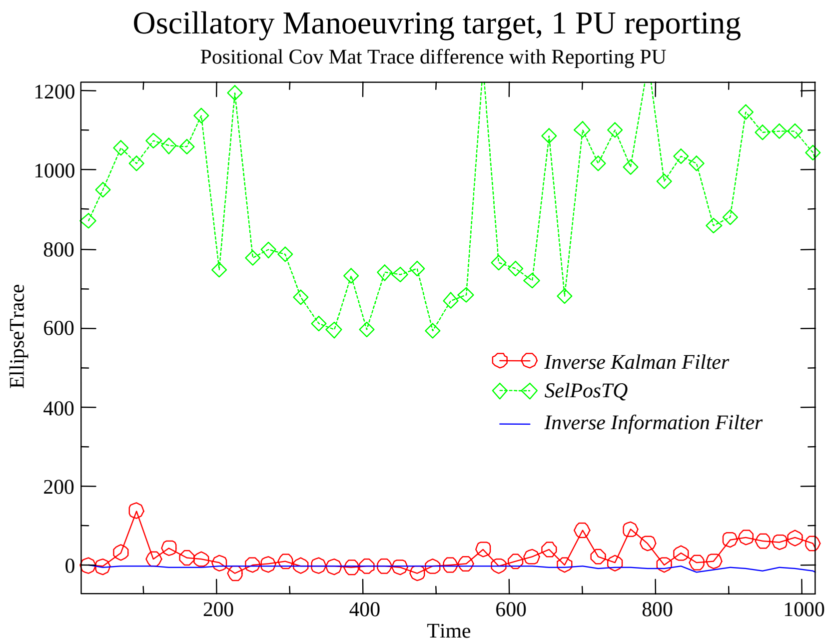



*Figure 28. Covariance matrix for manoeuvring target with 1 PU reporting*

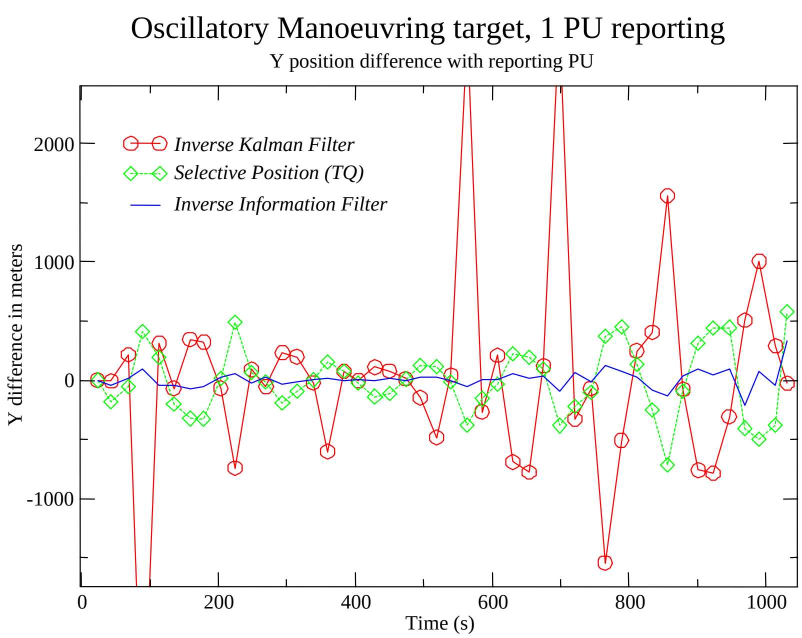



*Figure 29. Position for manoeuvring target with 1 PU reporting*

### 8.1.2 Test 2: two PUs reporting

The two-PUs-reporting scenario has two reporting units, both observing all targets with their sensors. Since the test-bed does not support the centralized fusion paradigm where a central fusion node fuses contact data from all fusion nodes, comparison with contact fusion is not possible. Instead, each track fusion method is compared with the others to assess its relative performance.

The methods compared are again the inverse Kalman Filter, the inverse Information Filter, the SPF with TQ and the SPF with covariance matrix (Cov Mat). Let's suppose that the contact fusion of two sources has a better precision than the fusion of one source only; then the SPF with Cov Mat can be used as an estimate of the upper bound of the contact fusion positional error. This approximation makes sense since the SPF with Cov Mat method returns the latest report's covariance matrix. Because the global node fuses two sources of data, it is expected that the overall result of the fusion has a lower positional uncertainty than SPF with Cov Mat. Then the best methods will be those that are lesser than SPF with Cov Mat but somehow follow the same pattern.

Figure 30 shows the trace of the covariance matrix for the Mig-21 and Figure 31 for the TU-160. As expected, the trace obtained from the SPF with TQ is larger, always greater than the SPF with Cov Mat. The inverse Information Filter is always lesser than SPF with Cov Mat, but often very close. The inverse Kalman Filter tends to be lesser than the inverse Information Filter, but close to it. However, some values for the inverse Kalman Filter are greater than the SPF with Cov Mat values.

The oscillations observed in all graphs are the consequence of the two reporting PUs sending their data at approximately 1-second intervals with a net polling cycle time of 20 seconds. Having two track updates within a small time-frame makes the tracker more confident in the resulting position. Then the 20-second time lapse is taken into account and this enlarges the error. This effect is observed for both the Mig-21 and the TU-160.

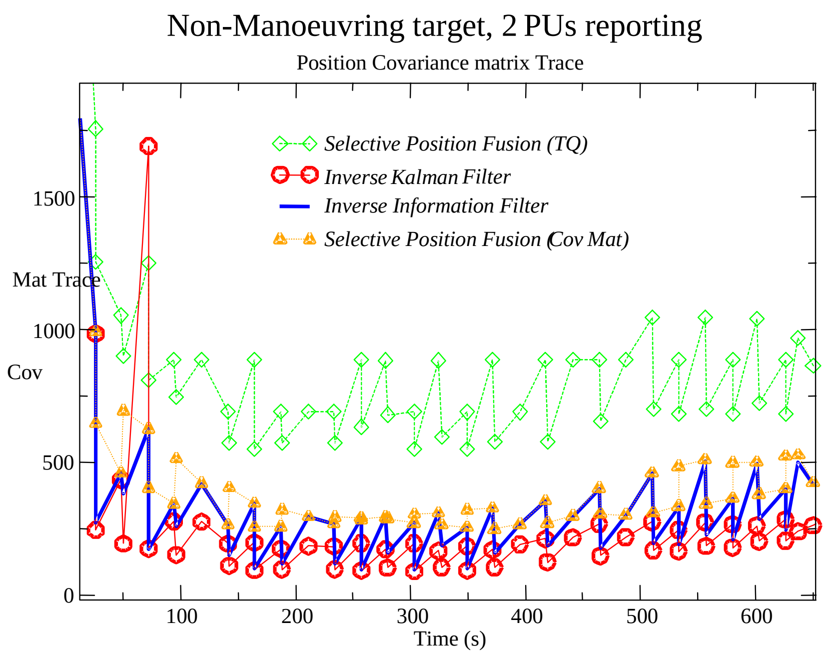



*Figure 30. Covariance matrix for non-manoeuvring target with 2 PUs reporting*

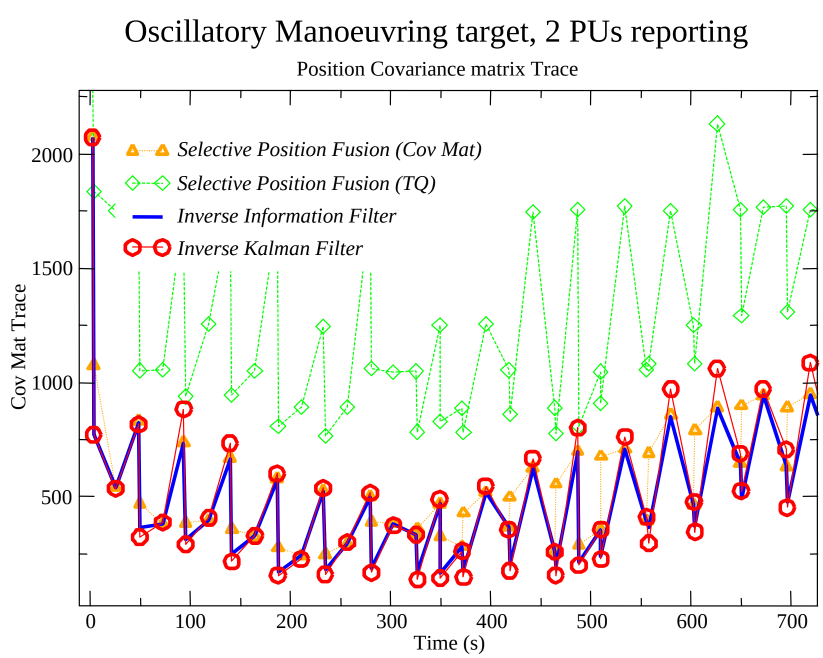



*Figure 31 Covariance matrix for oscillatory manoeuvring target with 2 PUs reporting*

### 8.1.3 Conclusion

Track fusion algorithms were tested and analyzed in the test-bed. The constraints of the test-bed force the Global MSDF of each PU to make no assumptions as to the configuration of the remote tracker that is reporting tracks. For example, the process noise used by the remote tracker is unknown when a reported track is received, so that information cannot be used during track fusion oat the receiving unit. For this reason, the process noise used by the inverse Information Filter is computed and used by the Global MSDF of the receiving unit.

From the above analysis of track fusion algorithms on manoeuvring and non-manoeuvring targets, the inverse Information Filter generally outperforms the other methods in the scenarios given. With one reporting unit, this method follows the results of the local track on the reporting unit. With two reporting units, the inverse Information Filter is as good as one of the reporting local trackers, which is given by the SPF with Covariance Matrix algorithm.

However, on a real system like Link-11, the full covariance matrix is not available for reported tracks. Therefore, none of the methods based on the covariance matrix can be used in this context. The only method left is the SPF with TQ. Although this method overestimates the error on the track, it does provide a consistent estimate of the position of a track. When only TQ information is available, the SPF with TQ method should be used.

Upon adding more sensors (like the SPS-49 and slaved IFFs) on each PU, the results become more complicated to interpret but the conclusions remain the same, with the Information Filter approach giving the best results. This conclusion also holds when comparing with centralized fusion results (Demers et al., 2003).

Finally, another track fusion algorithm that could be investigated further is the Covariance Intersection. This algorithm was described in Section 2.2.2.

## 8.2 Enhanced imaging capabilities

The FLIR is a passive detection system deployed on maritime surveillance aircraft (such as the CP-140 Aurora). It is used primarily for short- to medium-range recognition and identification of surface ships, which allows target verification and identification in daylight or total darkness. Such a detection system has been implemented in the test-bed.

A stand-alone FLIR image classifier was added to the test-bed. A special effort was made here to automate the complete process and render it real-time, unlike the FLIR classifiers used (and fused) in the second in a series of three DRDC-V reports on demonstrations of data/information concepts for the CP-140 (Valin et al., 2004a). Those classifiers required manual thresholding of imagery and off-line determination of attributes.

The classifier is used to enhance the identity information of a target through the classification of a FLIR image. The implementation of a FLIR simulator was deemed too prohibitive to develop, so attention was given to the classification algorithm. These simulators are costly and hard to come by. Instead, the database of Park and Sklansky (1990) was used to generate target images. These 2,545 images were provided by the United States Naval Air Warfare Center (NAWC) through Dr. Sklansky of the University of California at Irvine.

When the operator asks for an image, a random image of the appropriate type is chosen from a subset of all images and fed to the classifier. The action of choosing the appropriate image from the database is the FLIR simulator. It does not take into account the ship's heading or its distance from the imaging platform. No atmospheric data are factored in, and sensors are not modelled. It basically gives an image representing a ship of the appropriate type, which is sufficient for the needs of this project.

The test-bed ship recognition algorithm for FLIR imagery requires a prior segmentation of the FLIR image to separate the ship object from the sea surround and background. Once the ship is extracted from the image, features are computed and target identification is achieved. However, the identification results are highly dependent upon the quality of the extracted ship object.

The task of binary segmentation is usually achieved using either thresholding algorithms (manual or automatic) or Common Texture Analysis (CTA) using the Grey Level Co-occurrence Matrix (GLCM). These methods suffer from several drawbacks. Thresholding algorithms are often manual (i.e., requiring an input from the operator) and do not perform well with noisy images. On the other hand, CTA methods are computationally expensive and still do not perform well with noisy images.

The next few sections show the results obtained with the Visual Perception-based Segmentation (VPS) algorithm (Heucke et al., 2000) for FLIR imagery segmentation, and compare them with other existing thresholding methods. Results are shown for the segmentation procedure and the ship recognition task. Next, the feature extraction process is presented, and finally the classifier itself.

### 8.2.1 Segmentation algorithm

The VPS algorithm was specifically designed for the task of foreground/background separation. In our context of application, the segmentation task consists of ship/background separation.

This algorithm is derived from psycho-visual experiments on the adaptability of the eye and its minimum perceptible contrast. It offers a simplified description of the main features of the human visual system: transformation of luminance into a psycho-visual stimulus, brightness adaptation, and the existence of a minimum perceptible contrast (Heucke et al., 2000).

Three different brightness stimuli need to be modelled: the fovea centralis, the object background and the surrounding background.

The fovea centralis is the observation field that is always focused on the object of interest by the accommodation of the eye. The brightness stimulus of the fovea centralis ($H_0$) can be simplified and computed as follows:

$$H_O(x,y) = \frac{1}{N_O} \sum_{m=-\frac{p_O}{2}}^{\frac{p_O}{2}} \sum_{n=-\frac{p_O}{2}}^{\frac{p_O}{2}} I(x-m, y-n)$$

where $I(x, y)$ represents the grey-level intensity of the pixel and $N_0$ is the number of pixels in the simulated fovea centralis ( $N_O = p_O^2$ ). The parameter $p_0$, which is the radius of the region of interest, is set to 7, which is a good value for all images in the database.

The object background ($H_B$) is the close neighbourhood of the fovea centralis that has a strong influence on the perception of the object. It is modelled and computed as follows:

$$H_B(x,y) = \frac{1}{N_B} \sum_{m=-\frac{p_B}{2}}^{\frac{p_B}{2}} \sum_{n=-\frac{p_B}{2}}^{\frac{p_B}{2}} \frac{I(x-m, y-n)}{\sqrt{(x-m)^2 + (y-n)^2}}$$

where $N_B \equiv p_B^2$ and $p_B \equiv 2p_O$.

Contrast ( $C(x,y)$ ) is the perceived difference in luminance between objects and background. It refers to the normalized difference between the brightness stimuli of the fovea centralis $H_o$ and the object background $H_B$, and is computed as follows:

$$C(x,y) = \frac{\left|H_O(x,y) - H_B(x,y)\right|}{H_B(x,y)}$$

The human eye has the ability to adapt to different luminances. The adaptation illuminance ($H_A$) is simplified and calculated as follows (Heucke et al., 2000)

$$H_A = 0.923 H_B(x,y) + 0.077 H_S$$

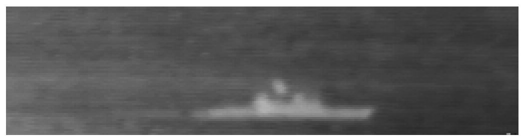
where $H_s$ is the surround background (the total intensity divided by the total number of pixels for the whole image). The minimum perceptible contrast is defined as the normalized amount of light that must be added to a brightness stimulus $H_O$ of the fovea centralis so that $H_O$ can just be discriminated from a reference field of the same brightness stimulus $H_B$. It is computed as follows:

$$C_{A,\min}(x,y) = \begin{cases} \dfrac{C_w}{H_B(x,y)} \left[0.808 + \sqrt{H_A(x,y)}\right]^2, \text{when } H_A \geq H_B \\ \dfrac{C_w}{H_B(x,y)} \left[0.808 + \sqrt{\dfrac{H_B(x,y)^2}{H_A(x,y)}}\right]^2, \text{when } H_A < H_B \end{cases}$$

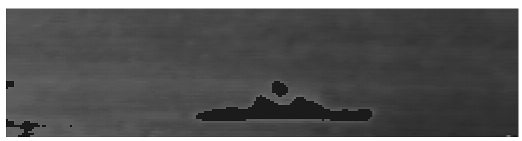
If $C$ is greater than $C_{A,\min}$ , a pixel belongs to a foreground object (the ship in our context of application) and $C_W$ is a constant which can be adjusted either manually, or using an automatic method (more research can be done to fully optimize the algorithm to adjust this

variable using some image features like the average grey level or the image entropy). Here it has been set to 2.2, which seems adequate for the vast majority of images.

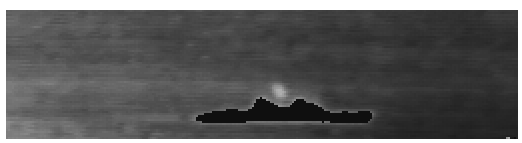
In noisy FLIR images (noise may be caused by fog, snow, etc.), some parts of the background might be misclassified as part of the foreground object (i.e., the target). Therefore, having multiple segmented objects in the resulting image, we need to find the object that truly represents the ship. To do so, we make the assumption that the brightest pixel of the scene should be part of the target. The segmented object that possesses this pixel is considered to be the target, and all the features required for ship recognition will be computed on that object. Figure 32 shows the effect of the post-processing algorithm. The first image is the original. In the second image, the visual perception-based segmentation is applied, but it can be seen that there are multiple segmented objects. Finally, in the last image, after post-processing the only remaining region is the one with the brightest pixel.



Original



VPS without post-processing



VPS with post-processing

*Figure 32. Effect of the post-processing step on the segmentation*

Results obtained using the VPS algorithm show a substantial improvement in ship segmentation compared with thresholding methods, particularly in noisy or low-contrast imagery. It can be seen that the segmented ship has a well-defined border and that the overall result contains fewer misclassified pixels. Results on different FLIR images are shown in Figure 33. These images show three segmentation procedures: manual thresholding, automatic thresholding and visual perception segmentation. The difference between the first two lies in the fact that the thresholding value, i.e., the value above which all pixels are considered part of the object, is chosen manually by the user in the first case and automatically in the second case. The automatic thresholding procedure shown here uses the scale-space algorithm.

The fact that this method is completely unsupervised makes it easy to incorporate into a fusion environment such as the test-bed.

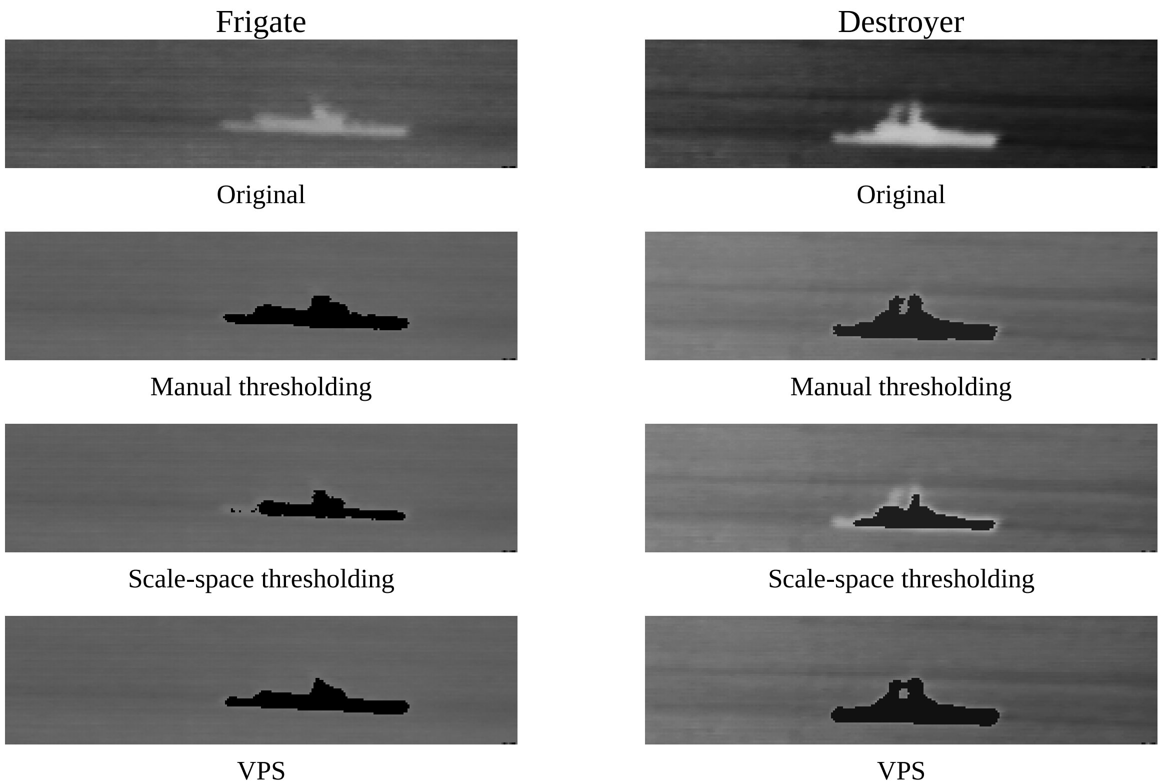



*Figure 33. Examples of segmentation: left a frigate, right a destroyer*

### 8.2.2 Feature extraction

Once the ship has been segmented, feature extraction can take place. Fourteen moments are used to describe the segmented image (Allen, 2001). Half of these moments are structural and the other half are intensity-based. The idea behind the use of intensity-based moments is to use all the information provided by an infrared image and not just shape information.

To compute the moments, the segmented ship is partitioned into seven equal sections along the *x*-axis, and into two sections along the *y*-axis delimited by a centroid point defined below.

The seven structural moments are computed on the part of the segmented ship that is above the centroid, i.e., on the discriminating part which is above the hull. The centroid used here takes into account the intensity of each pixel and is defined as:

$$\bar{x} = \frac{1}{\sum_k (I_k - c)} \sum_l x_l (I_l - c)$$

$$\bar{y} = \frac{1}{\sum_k (I_k - c)} \sum_l y_l (I_l - c)$$

where $I_l$ is the grey-level intensity of pixel $l$ and $c$ is a constant. The following moment

$$\mu^i = \frac{1}{LN_i^+} \sum_{l \in S_i^+} (y_l - \bar{y})$$

is computed for each of the seven sections of the segmented ship. Here *i* is the ship section number, *L* is the ship length in pixels, and *N* is the number of pixels in $S_i^+$, the *i*-th of the seven sections above the centroid point. These are the structural moments.

The seven intensity-based moments are given by

$$\nu^i = \frac{1}{\sum_l (I_l - c)} \sum_{l \in S_i^+} (I_l - c) .$$

For each section *i*, the intensity values of all pixels are summed and divided by the ship's overall intensity.

### 8.2.3 Classification using moments of the segmented image

By computing moments over a test sample of segmented ships, a knowledge database is built from which frequency distributions are extracted (Demers, 2003). These frequency distributions are used to compute a probability of occurrence for each combination of moments and classes of ships of an unknown feature vector. From these probabilities of occurrence, expert opinions in the form of proposition sets are computed where a particular proposition corresponds to the probability that a given moment represents a given class. Given an unknown feature vector, 14 moments combined with eight classes gives 14 proposition sets of eight propositions each. The Dempster-Shafer theory of evidence is used to combine the 14 expert opinions. The output is a belief score for each ship class. The eight classes to be determined are: destroyer (DT), container (CT), civilian freighter (CF), auxiliary oil replenishment (AOR), landing assault tanker (LAT), frigate (FR), cruiser (CR) and destroyer with guided missile (DGM).

The confusion matrix of the moment-based classifier is given in Table 6. The table represents the accuracy defined as the number of ships correctly classified over the total number of ships of that category. For some ship types, the performance is very poor. The True Acceptance Rate (TAR) is given as the number of ships correctly classified in a given category over the total number of ships misclassified in that category. It represents the confidence that the output really is what the classifier claims it is.

*Table 6. Confusion matrix for the moment-based classifier in percentages with TAR = 75.5%.*

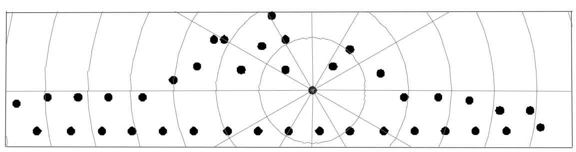
| | DT | CT | CF | AOR | LAT | FR | CR | DGM |
|---|---|---|---|---|---|---|---|---|
| DT | **78.9** | 3.6 | 2.1 | 0.4 | 1.5 | 6.0 | 5.2 | 2.4 |
| CT | 4.9 | **79.2** | 11.5 | 2.2 | 0.9 | 0.9 | 0.4 | 0.0 |
| CF | 1.6 | 12.0 | **85.3** | 0.5 | 0.5 | 0.0 | 0.0 | 0.0 |
| AOR | 1.7 | 2.2 | 11.7 | **84.4** | 0.0 | 0.0 | 0.0 | 0.0 |
| LAT | 14.4 | 5.3 | 9.0 | 1.1 | **67.0** | 2.7 | 0.0 | 0.5 |
| FR | 27.2 | 9.8 | 4.9 | 0.0 | 0.0 | **50.0** | 7.1 | 1.1 |
| CR | 12.7 | 0.6 | 0.3 | 0.6 | 0.3 | 8.9 | **76.3** | 0.3 |
| DGM | 24.2 | 1.7 | 2.1 | 0.8 | 1.3 | 5.1 | 1.7 | **63.1** |
| TAR | 75.3 | 66.1 | 56.1 | 95.9 | 87.5 | 50.0 | 80.9 | 87.1 |

### 8.2.4 Template-based classification

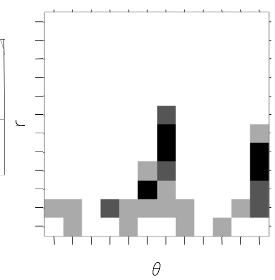
A very different classifier based on templates is chosen to complement moment-based classification. It uses a shape-matching algorithm to compute a distance from each template and thus classify an unknown ship. To achieve this, a contour is extracted from the segmented image and shape descriptors are calculated on this contour. Using those shape descriptors, points of the unknown ship are assigned to corresponding points of a template and a mapping transformation is computed from this assignment. This transformation is used to achieve translation, rotation and scale invariance. A distance measure between the unknown input and each available template is calculated and used as a classification basis. The shape descriptor and matching algorithm used were detailed by Belongie (2002) and are briefly described below.

Feature extraction here consists of computing shape descriptors (also called shape contexts) that will be used for template matching. In this approach, a shape is essentially captured by a subset of points of the external and/or internal contours. The subset is chosen as uniform spaced points of the contour, i.e., a down-sampled contour. A shape context is computed for each point of the subset. The ensemble of shape contexts represents the reduced object information needed for matching.

A shape context corresponding to a point $p_i$ is the spatial distribution of all other points $q_i$ about point $p_i$. In other words, for a point $p_i$ on the shape, a coarse histogram $h_i$ of the relative coordinates of the remaining $n$–1 points is computed,



(a)




(b)

.

*Figure 34. Shape context computation*

An example is shown in Figure 34 (Demers, 2003) where (a) shows overlaid histogram bins used in computing the shape context of the centre point and (b) gives a graphical representation of the shape context $r(\theta)$ associated with the centre point of (a).

The cost of matching two points—$p_i$ from the first shape and $q_i$ from the second shape—is denoted $C_{ij} = C(p_i,q_j)$. The $\chi^2$ test statistic is used

$$C_{ij} = \frac{1}{2}\sum_{k=1}^{K}\frac{\left[h_i(k)-h_j(k)\right]^2}{h_i(k)+h_j(k)}$$

where $h_i(k)$ and $h_j(k)$ denote the $K$-bin normalized histogram at $p_i$ and $q_j$, respectively.

$$h_i(k) = \#\{q \neq p_i : (q - p_i) \in \text{bin}(k)\}$$

The assignment problem is solved by minimizing the total cost of matching, given a set of costs $C_{ij}$ between all pairs of points $p_i$ on the first shape and $q_j$ on the second shape, subject to the constraint that the matching be one-to-one, namely $\pi(i)$ is a permutation.

$$H(\pi) = \sum_i C(p_i, q_{\pi(i)})$$

This is a linear assignment problem solved by the Jonker-Volgenant-Castanon algorithm (Drummond et al., 1990). The input is a square cost matrix with entries $C_{ij}$, and the result is a permutation assigning one point $p_i$ of the first shape to one and only one point $q_i$ of the second shape.

Having a one-to-one correspondence between two shapes, a plane transformation is estimated to map arbitrary points from one shape to the other. By mapping all points of a template onto the unknown ship, translation, rotation and scaling invariance are achieved. A function known as a thin plate spline is used to estimate the coordinate transformation. This is the 2D generalization of the cubic spline (see Bookstein (1989) for further details). With the template transformed onto the unknown ship, it is possible to estimate a distance measure between the two. The measure used is the symmetric sum of the minimum Euclidian distance over a subset of the best matching points of two shapes $P$ and $Q$, i.e.

$$D_{sc}(P,Q) = \frac{1}{n}\sum_{p\in P} \arg\min_{q\in Q} \left\| p - T(q) \right\| + \frac{1}{m}\sum_{q\in Q} \arg\min_{p\in P} \left\| p - T(q) \right\|$$

where $T(q)$ denotes the estimated thin plate spline shape transformation of point $q$, and $n$ and $m$ are the number of points of shape $P$ and $Q$, respectively. Classification is achieved by calculating this distance for each template to the unknown ship. The smallest distance indicates that the corresponding template is the most likely class candidate for the unknown input.

The overall accuracy of the template-based classifier is lower than the moment-based one, 73.1% compared to 75.5% (see Table 7). Although the overall appearance of the confusion matrix resembles the previous one, closer inspection shows that in many cases, complementarity is achieved. Analysis of per-category performance indicates that both classifiers exhibit the same behaviour with increasing ship distance.

*Table 7. Confusion matrix for the template-based classifier in percentages with TAR = 73.1%.*

| | DT | CT | CF | AOR | LAT | FR | CR | DGM |
|---|---|---|---|---|---|---|---|---|
| DT | **60.3** | 2.1 | 0.7 | 4.9 | 2.9 | 7.4 | 9.4 | 12.2 |
| CT | 0.0 | **95.6** | 0.9 | 0.4 | 3.1 | 0.0 | 0.0 | 0.0 |
| CF | 1.1 | 31.7 | **51.9** | 12.0 | 3.3 | 0.0 | 0.0 | 0.0 |
| AOR | 0.0 | 0.0 | 0.2 | **99.8** | 0.0 | 0.0 | 0.0 | 0.0 |
| LAT | 5.3 | 2.7 | 0.5 | 13.8 | **72.9** | 2.1 | 1.1 | 1.6 |
| FR | 5.4 | 3.3 | 0.0 | 3.3 | 6.5 | **61.4** | 9.8 | 10.3 |
| CR | 7.3 | 0.3 | 0.0 | 1.3 | 7.3 | 12.1 | **69.5** | 2.2 |
| DGM | 12.3 | 0.0 | 0.0 | 3.0 | 6.4 | 1.3 | 1.7 | **75.4** |
| TAR | 86.0 | 71.5 | 91.3 | 80.2 | 61.7 | 52.8 | 69.7 | 59.5 |

The output of the template-based classifier is a distance measure which must be transformed into a pseudo-confidence level for further processing, e.g., by Dempster-Shafer evidential reasoning. The following function was used to map the output of a distance function to the range [0,1]:

$$f(x) = e^{-x^2}$$

Following on the work of Xu et al. (1992), information from the confusion matrix of the test sample was used to weight the output of each classifier. The confidence measure assigned to class label $C_i$ was weighted by the corresponding TAR.

### 8.2.5 Fusion of classifiers

Ho (2002) pointed out two strategies for designing multiple classifier systems, namely the decision optimization methods and the coverage optimization methods. In the former case, the classifiers are given and unchangeable, and are considered as already optimized for the task, even though they might not be. The goal is then to optimize the combination function to make the best of what the classifiers give. In the case of coverage optimization methods, the combination function is fixed and unchangeable in form. The strategy is to create a set of classifiers that will complement each other and will yield the best final decision under the chosen combination function.

In the same paper Ho summarized the best known decision combination methods under joint consideration of two factors: the possibility of training the classifiers and the level of information provided by the classifiers, i.e., unique, ranked or measurement level. Another consideration for the experiment described in this paper would be the number of classifiers forming the ensemble. As that number diminishes, the optimization focus must shift from the combination function to the coverage, because relying on a statistical combination of poorly performing classifiers is not possible in that case. With a small number of classifiers the impact of one being wrong on the final decision is substantial. Hence, the performance of each individual classifier must be optimal, and uniform coverage of the input patterns must be achieved. Furthermore, to ensure accuracy improvement with a small number of classifiers, the secondary choice of each one has to be considered. In the FLIR experiment, when both classifiers are wrong but their secondary choices are right, the combination scheme must be able to generate the correct final decision. This is possible when the output of each classifier is a confidence or distance measure of some sort and the combination function takes all outputs into consideration.

Two combination schemes were investigated:

1. the product rule and
2. the Dempster-Shafer theory of evidence.

The product rule of combination quantifies the likelihood of the input pattern being assigned a class label $C_i$ by combining the *a posteriori* probabilities generated by each individual classifier for that particular class label.

Consider a pattern recognition problem where $R$ classifiers are to give their opinion on the possible membership of an unknown input pattern $x$ in $M$ classes. Under the condition of statistical independence, the Bayesian decision rule can be modified into the product decision

rule (Kittler et al., 1998). This rule states that $x$ is assigned to class $C_i$ provided that the following is maximum:

$$\max_{i=1}^{M}\left[ P^{-(R-1)}(C_i)\prod_{k=1}^{R} P_k(x\in C_i)\right]$$

where $P(C_i)$ is the *a priori* probability of occurrence and the $P_k$ are the *a posteriori* probabilities. The condition of independence here is a good approximation considering that each classifier uses its own internal representation of the input pattern. The weighted confidence levels of each classifier are taken as *a posteriori* probabilities.

The confusion matrix for the product fuser is shown in Table 8. The overall accuracy of the product fuser is 80.8%. An improvement of 5% over the best individual classifier is observed.

*Table 8. Confusion matrix for the product fuser, with TAR = 80.8%.*

| | DT | CT | CF | AOR | LAT | FR | CR | DGM |
|---|---|---|---|---|---|---|---|---|
| DT | **80.0** | 3.2 | 0.8 | 2.8 | 0.5 | 1.9 | 7.6 | 3.3 |
| CT | 0.0 | **91.2** | 4.9 | 3.1 | 0.9 | 0.0 | 0.0 | 0.0 |
| CF | 0.5 | 18.6 | **76.0** | 4.9 | 0.0 | 0.0 | 0.0 | 0.0 |
| AOR | 0.0 | 1.0 | 4.5 | **94.5** | 0.0 | 0.0 | 0.0 | 0.0 |
| LAT | 12.8 | 2.1 | 6.4 | 4.8 | **72.9** | 0.0 | 0.0 | 1.1 |
| FR | 23.9 | 4.3 | 1.6 | 2.7 | 0.5 | **56.5** | 9.2 | 1.1 |
| CR | 7.0 | 0.3 | 0.0 | 1.0 | 1.0 | 6.0 | **83.8** | 1.0 |
| DGM | 22.0 | 0.8 | 0.4 | 0.8 | 0.4 | 0.8 | 0.4 | **74.2** |
| TAR | 80.8 | 72.8 | 72.8 | 87.6 | 92.6 | 74.8 | 77.9 | 84.5 |

The Dempster-Shafer theory of evidence has been used previously to combine the output of multiple classifiers (Xu, 1992). The technique is more robust than Bayesian approaches because the uncertainty of each classifier is taken into account, although a test sample is necessary to determine it.

The masses or basic probability assignments are taken from the weighted measures of each classifier. The ignorance is taken as the accuracy of each classifier for the particular ship category under consideration. This differs from the approach of Xu, where the classifiers' output is of the abstract type. Here, all the information provided by the classifiers is used. The confusion matrix for the DS fuser is shown in Table 9. The overall accuracy of the product fuser is 80.5%. An improvement of 5% over the best individual classifier is again observed.

*Table 9. Confusion matrix of the Dempster-Shafer fuser, with TAR = 80.5%.*

| | DT | CT | CF | AOR | LAT | FR | CR | DGM |
|---|---|---|---|---|---|---|---|---|
| DT | **80.0** | 2.5 | 1.1 | 3.4 | 0.4 | 1.9 | 8.0 | 2.8 |
| CT | 0.9 | **89.4** | 4.4 | 3.5 | 1.8 | 0.0 | 0.0 | 0.0 |
| CF | 0.5 | 19.1 | **75.4** | 4.9 | 0.0 | 0.0 | 0.0 | 0.0 |
| AOR | 0.0 | 0.5 | 2.2 | **97.5** | 0.0 | 0.0 | 0.0 | 0.0 |
| LAT | 12.8 | 1.6 | 7.4 | 5.9 | **70.2** | 0.5 | 0.0 | 1.6 |
| FR | 27.2 | 4.9 | 2.7 | 2.7 | 0.5 | **50.0** | 11.4 | 0.5 |
| CR | 7.3 | 0.3 | 0.0 | 2.2 | 0.3 | 3.5 | **85.1** | 1.3 |
| DGM | 20.8 | 1.3 | 0.4 | 0.8 | 0.8 | 1.3 | 0.8 | **73.7** |
| TAR | 80.2 | 72.7 | 74.6 | 85.7 | 91.3 | 76.0 | 76.4 | 85.7 |

### 8.2.6 Performance evaluation

Performance is evaluated in the context of a completely automated process. This constraint is a very stringent one, particularly for the segmentation part. However, it permits an objective study of a complete reconnaissance system and the interaction of its component parts.

The segmentation algorithm has to deal with images having a broad range of signal-to-noise ratios (SNR). The parameters of the algorithm have been adjusted for a typical image. No automatic adjustment of the segmentation based on general characteristics of the image has been implemented and no image pre-processing is done. As the ship image gets smaller (i.e., the distance increases between observer and ship), the SNR diminishes and discriminating features disappear. This effect cannot be overcome. However, the lower SNR can be compensated by an appropriate segmentation and pre-processing. This would yield a significant performance improvement.

The moment-based classifier uses features based on the brightness of the pixel. The analysis of the probability of occurrence shows that the discriminative power of these attributes is not significant. However, the low resolution of some images in the database preclude their use. High-resolution images should be tested. The use of Zernike's moments, Hu's moments and other geometric moments should be investigated. Along with new moments, the implementation of a feature selection algorithm would also be beneficial. This not only reduces the burden of the recognition process by reducing the number of features, but in some cases it can also provide better classification accuracy (Raudys and Jain, 1991)

Analysis of the accuracy per number of pixels on the segmented ship shows that the template-based classifier generally performs better for ships near the observer, i.e., with a large number of pixels. This is caused mainly by the choice of templates. They were built from high-definition images with many details. A multi-resolution set of templates could improve performance, particularly if the distance to the ship is known. This would generally be the case if the ship is already tracked by radar.

Regarding the template matching process, a circular order preserving algorithm was implemented, but deemed too CPU-intensive for the added benefits. This algorithm preserves the circular order of points on both templates, which helps deal with outliers.

This experiment was conducted on images of ships viewed from a 90° angle at sea level. In a real-life scenario any angle would be possible. Both classifiers would have to be modified to take into account a change in angle lookdown (in the case of an airborne imager) and relative heading.

Even though the product rule of combination performs slightly better, the Dempster-Shafer rule of combination is the preferred method here. It is more robust in the sense that the state of unknown information is represented by the ignorance.

It is clear from this study that the combining function is not the primary objective of optimization. The suite of classifiers must be improved first. It is a case of coverage optimization, not combining function optimization.

### 8.2.7 Conclusion

An automatic recognition system using two classifiers has been presented which classifies FLIR ship images viewed from 90° into eight categories. A moment-based classifier and a

template-based classifier provide confidence measures to a fuser. Two combination functions were studied: the product rule of combination and the Dempster-Shafer combination method. The Dempster-Shafer fuser achieved an overall classification accuracy of 80.5%, which is better than the best individual classifier.

This experiment demonstrates that with a small number of classifiers, each must output a secondary choice with an associated confidence level. The combining function must use all this information in order to improve results. Accuracy improvement comes primarily from the optimization of the ensemble of classifiers and not from the combining function.

# 9. Conclusions and recommendations

The main objective of this report was to analyze methods, techniques, algorithms, rule-based communication protocols and infrastructures needed to establish a Maritime Tactical Picture (MTP). An MTP is, by definition, the combination of the Local Area Picture (LAP) seen by a unit (which may be part of a task force) using its own sensors and a Wide Area Picture (WAP) using information, not controlled by the task force, provided by high frequency (HF) or ultra high frequency (UHF) radios or satellites. For that purpose a test-bed was developed to test and benchmark all the above-mentioned elements.

This report focused on:

- Communication and information exchange for a simplified MTP composed of the combination of the LAP as seen by several units (HALIFAX class frigate and aircraft such as the CP-140 Aurora) cooperating in a task force

- Algorithmic requirement analysis and algorithm development and enhancement for LAP and WAP establishment, including imaging and non-imaging information, especially an automated classifier for infrared imagery
- Simulation environments and architectures that are best suited for scenario generation and sensor simulation for multiple collaborating multi-sensor Command and Control Systems (CCSs), and the capability to provide a fusion test-bed as a modular entity to larger High Level Architecture (HLA) operational federations.

This test-bed provided the tools to study what information should be communicated, when and how, and the impact it has on joint situational awareness.

The selected approach for the implementation of the track-level fusion system was based on a powerful real-time KBS based on the BB paradigm, which supports real-time distributed processing. This KBS BB, called Cortex (developed at Lockheed Martin Canada in collaboration with DRDC Valcartier), allows multiple blackboards to run on multiple machines with integrated communications. These features permit a highly configurable and flexible design for track-level design. Three blackboards were needed, one for the Local MSDF fusing local sensor data, one for the Global MSDF fusing remote information, and one for Information Management. Information prioritization followed the Canadianization of Handbook 5 recommendations.

A survey of simulation environments was also performed and partial HLA compliance was demonstrated on the test-bed.

Results have shown that the inverse Information Filter provides the best approach for track-to-track fusion when full covariances are available from the PUs. However, on a real existing system like Link-11, the full covariance matrix is not available for reported tracks. The only method that is left in that case is Selective Position Fusion with Track Quality. Although this method overestimates the error on the track, it does provide a consistent estimate of the position of a track.

An automated FLIR system was designed and implemented that fuses the results of two classifiers in a manner reminiscent of the DRDC Valcartier technical report (TR 2004-282)

entitled “A Survey of Information Fusion Algorithms for an Airborne Application”. That report fused four classifiers by two methods with better performance, but some off-line processing was needed, precluding complete automation.

# 10. References


Allen, S. (2001). Signal Based Features with Application to Ship Recognition in FLIR Imagery, in Proceedings of the 4th Annual Conference on Information Fusion, FUSION 2001, vol. 2, pp. FrC2-3 to FrC2-8, Montréal, Canada, August 7-10, 2001.

Belongie, S., Malik J. & Puzicha, J. (2002). Shape Matching and Object Recognition Using Shape Contexts", IEEE PAMI, vol. 24, no. 24, pp. 509-522, Apr 2002.

Benaskeur A. R. & Roy, J. (2002). A comparative study of data fusion algorithms for target tracking, DRDC Technical report # TR 2001-224, October 2002, 75 pages.

Bookstein, F. L. (1989). Principal Warps: Thin-Plate Splines and the Decomposition of Deformations, IEEE PAMI, vol. 11, no. 6, pp. 567-585, Jun 1989.

Carlotto, M. J. (1987). Histogram Analysis Using a Scale-Space Approach, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 9, No.1, pp. 121-129, January 1987.

Chen, H. , Kirubarajan, T. & Bar-Shalom, Y. (2000). Comparison of Centralized and Distributed Tracking Algorithms Using Air-to-Air Scenarios, in Proceedings of Signal Processing of Small Targets 2000, SPIE vol. 4048, pp. 440-451.

Chong, C.Y., Mori, S. & Chang, K.C. (1998). Distributed Multitarget Multisensor Tracking, Chapter 8 in MultiTarget Multisensor Tracking: Advanced Applications, Y. Bar-Shalom Ed., ISBN 0964831228.

Demers, H. (2003). Fusion of Two Imagery Classifiers: a Case Study, in Proceedings of NATO Advanced Study Institute (ASI) on "Data Fusion for Situation Monitoring, Incident Detection, Alert and Response Management", Tsakhkadzor, Armenia, 18-29 August 2003, E. Shahbazian, G. Rogova, and P. Valin, editors (to be published by IOS Press).

Drummond, O. E., Castanon, D.A. & Bellovin, M.S. (1990) Comparison of 2-D Assignment for Sparse, Rectangular, Floating Point, Cost Matrix, Journal of the SDI Panels on Tracking, Institute for Defense Analyses, no.4, pp.4-81 to 4-97, 1990.

Drummond, O. E. (2001). Track and Tracklet Fusion Filtering Using Data from Distributed Sensors, Proceedings of the Workshop on Estimation, Tracking and Fusion: A Tribute to Yaakov Bar-Shalom, pp. 167-186, May 2001.

Heucke, L., Knaak, M. & Orglmeister, R. (2000). A New Image Segmentation Method Based on Human Brightness Perception and Foveal Adaptation, IEEE Signal Processing Letters, Vol. 7, No. 6, pp.129-131, June 2000.

Ho, T. K. (2002). Multiple Classifier Combination: Lessons and Next Steps, in Hybrid Methods in Pattern Recognition, eds. A. Kandel, H. Bunke, 2002.

Hu, M. K. (1962). Visual pattern recognition by moment invariants, IEE Trans. Inform. Theory IT-8, pp 179-187, 1962.

Jarry, B., Jouan A. & Michalska, H. (2000). An IMM-JVC algorithm for target tracking with asynchronous sensors, Proceedings of the IASTED International Conference on Modeling, Identification and Control, pp. 603-607, Innsbruck, Austria, February 14-17, 2000.

Kittler, J., Duin, R.P.W. & Matas, J. (1998). On Combining Classifiers, IEEE PAMI, vol. 20, no. 3, pp. 226-239, Mar 1998.

Kuhl, F., Weatherly, R. and Dahmann, J. (1999). Creating Computer Simulation Systems: An Introduction to the High Level Architecture, Prentice-Hall PTR, Upper Saddle River, NJ, ISBN:0-13-022511-8, 212 pages, 1999.

Llinas, J. & Hall, D. (2001). Handbook of Multisensor Data Fusion, CRC Press, ISBN-08493-2379-7, 568 pages, 2001.

Lockheed Martin Canada, (2003). Command Decision Aid Technology I (COMDAT I) Cycle II Technology Demonstrator (TD) Software Design Document (SDD), CSES Task 0207, 6520207018 Rev. 1, 31 March 2003.

Lockheed Martin Canada, (1999). Canadianization of Handbook 5, CSIS task 109, 500109001 Rev. 1, 22 June 1999

Malmberg,A. & Karlsson, M. (2000). Track-to-Track Association in Decentralized Tracking Systems with Feedback, in Proceedings of Signal Processing of Small Targets 2000, SPIE vol. 4048, pp. 461-472.

McMichael, D. & Karan, M. (1999). Optimal Track Fusion using Bayes Factors, in Proceedings of  Information Decision and Control, Adelaide, Australia, pp. 117-122, 1999.

Ménard, E., Turgeon, D., Michaud, G. & Shahbazian E. (2003), in Proceedings of NATO Advanced Study Institute (ASI) on "Data Fusion for Situation Monitoring, Incident Detection, Alert and Response Management", Tsakhkadzor, Armenia, 18-29 August 2003 (to be published by Kluwer Academic Publishers).

Nicholson, D. & Deaves, R. (2000). Decentralized Track Fusion in Dynamic Networks, in Proceedings of Signal Processing of Small Targets 2000 , SPIE vol. 4048, pp. 452-460.

Nicholson, D., Julier, S.J. & Uhlmann, J.K. (2001). DDF: An evaluation of Covariance Intersection, in Proceedings of the 4th Annual Conference on Information Fusion, FUSION 2001, Montréal, Canada, 7-1- August 2001, pp. WeA2-19 to WeA2-25.

Park, Y. & Sklansky, J. (1990). Automated design of linear tree classifiers, Pattern Recognition, Vol. 23, No. 12, pp 1393-1412, 1990.

Peters, D. J. (2002). The Impact of Treating Tracks as Contacts in Kinematic Data Fusion, Proceedings of the International Command and Control Research and Technology Symposium, September, 2002.

Raudys, S.J. & Jain, A.K. (1991). Small Sample Size Effects in Statistical Pattern Recognition: Recommendations for Practitioners, IEEE PAMI, vol. 13, no. 3, pp. 252-264, Mar 1991.

Rothrock, R.L. & Drummond, O.E. (2000). Performance Metrics for Multiple-Sensor, Multiple-Target Tracking, in Proceedings of Signal and Data Processing of Small Targets 2000, vol. 4048, pp. 521-531.

Tremblay, C. & Valin, P. (2002). Experiments on individual classifiers and on fusion of a set of classifiers, in Proceedings of FUSION 2002, Annapolis, MD, July 8-11, 2002, pp. 272-277.

Tremblay, C. & Valin, P. (2003). Dempster-Shafer classifiers for FLIR imagery and neural net fusion of complementary classifiers, in Proceedings of NATO RTO Sensors and Electronics Technology (SET-059) Panel Symposium on “Target Tracking and Sensor Data Fusion for Military Observation Systems”, Budapest, Hungary, October 15-17, 2003.

Utete, S. (1998). Local Information Processing for Decision Making in Decentralized Sensing Networks, Lecture Notes In Computer Science; Vol. 1415, Springer-Verlag, Proceedings of the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems: Methodology and Tools in Knowledge-Based Systems, June 1-4, 1998, pp. 667-676.

Valin, P., Couture, J. & Simard, M.-A. (1996). Position and attribute fusion of radar, ESM, IFF and DataLink for AAW missions of the Canadian Patrol Frigate, Proceedings of Multisensor Fusion and Integration for Intelligent Systems (MFI ’96), Washington D.C., December 8-11, pp 63-71, 1996.

Valin, P., Bossé, E. & Jouan, A. (2004). Information Fusion Concepts for Airborne Maritime Surveillance and $C^2$ Operations, DRDC-V TM-2004-281.

Valin, P., Bossé, E. & Jouan, A. (2004). A Survey of Information Fusion Algorithms for an Airborne Application, DRDC-V TR-2004-282.

Valin, P., Jouan, A. & Bossé, E. (2004). Demonstration of Data/Information Fusion Concepts for Airborne Maritime Surveillance Operations, DRDC-V TR-2004-283.

Xu, L., Krzyzak, A., & Suen, C.Y. (1992). Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition, IEEE SMC, vol. 22, no. 3, pp. 418-435, May/June 1992.

# Acronyms

| | |
|---|---|
| ADP | Application Demonstration Platform |
| AFLC | Adaptive Fuzzy Logic Correlator |
| ASCACT | Advanced Shipboard Command and Control Technology |
| BB | BlackBoard |
| BIF | Bearing Intercept Fix |
| C2 | Command and Control |
| C4I | Command, Control, Communications, Computers, and Intelligence |
| CASE-ATTI | Concept Analysis and Simulation Environment for Automatic Target Tracking and Identification |
| CCIS | Command and Control Information System |
| CCS | Command and Control System |
| CCTT | Close Combat Tactical Trainer |
| CEC | Cooperative Engagement Capability |
| CGF | Computer Generated Forces |
| CI | Covariance Intersection |
| CPF | Canadian Patrol Frigate |
| CPU | Central Processing Unit |
| CSIS | Combat System In-Service Support |
| CVCA | Constant-Velocity Constant-Acceleration |
| DARPA | Defense Advanced Research Projects Agency |
| DIF | Data Interchange Format |
| DMSO | Defense Modeling and Simulation Office |
| DTDMA | Dynamic TDMA |
| EAKF | Extended Adaptive Kalman Filter |
| ESM | Electronic Support Measures |
| ExportCGF | Export Computer Generated Forces |
| FED | Federation Execution Data |
| FLIR | Forward Looking Infrared |
| FOM | Federation Object Model |
| FPU | Forwarding Participating Unit |
| FRU | Forwarding Reporting Unit |
| GCCS | Global Command and Control System |
| GCCS-M | GCCS-Maritime |
| GTDB | Global Track Data Base |
| HF | High Frequency |
| HLA | High Level Architecture |
| IEEE | Institute of Electrical and Electronic Engineers |
| IFF | Identification Friend or Foe |
| IM | Information Management |
| IMM | Interacting Multiple Model |
| ITAR | International Traffic in Arms Regulations |
| JPDA | Joint Probabilistic Data Association |

| | |
|---|---|
| JVC | Jonker-Volgenant-Castanon |
| KBS | Knowledge-Based System |
| LAMPS | Light Airborne Multi-Purpose System |
| LAP | Local Area Picture |
| LM | Lockheed Martin |
| MHT | Multi-Hypotheses Tracking |
| M&S | Modelling and Simulation |
| ModSAF | Modular Semi-Automated Forces |
| MOP | Measure of Performance |
| MSDF | Multi-Source Data Fusion |
| MTP | Maritime Tactical Picture |
| NCS | Net Control Station |
| NCW | Network-Centric Warfare |
| NILE | NATO Improved Link Eleven |
| NN | Nearest Neighbour |
| NU | NILE Unit |
| OMT | Object Model Template |
| ORTT | Operations Room Team Trainer |
| PLI | Participant Location and Identification |
| PU | Participating Unit |
| QAB | Quick Action Button |
| $R^2$ | Reporting Responsibility |
| RI | Runtime Infrastructure |
| RM | Resource Management |
| RTD | Run-Time Display |
| SAF | Semi-Automated Forces |
| SADM | Ship Air Defence Model |
| SAM | Surface-to-Air Missile |
| SEATS | Simulation Environment for the Analysis of the Tactical Situation |
| SIMNET | Simulator Networking |
| SNR | Signal-to-Noise Ratio |
| Soar-IFOR | Soar Intelligent FORces |
| SOM | Simulation Object Model |
| STA | Situation Threat Assessment |
| S-TADIL J | Satellite Tactical Data Information Link J |
| STAGE | Scenario Toolkit And Generation Environment |
| STDL | Satellite TDL |
| STRICOM | Simulation, Training and Instrumentation Command |
| TAR | True Acceptance Rate |
| TCP/IP | Transmission Control Protocol/Internet Protocol |
| TDL | Tactical Data Link |
| TDMA | Time Division Multiple Access |
| TDS | Tactical Data System |
| TSA | Tactical Situation Analysis |
| TQ | Track Quality |
| UHF | Ultra High Frequency |

| | |
|---|---|
| UWW | Under Water Warfare |
| VMF | Variable Message Format |
| WAP | Wide Area Picture |

# Annex

## Main sources of documentation

Since information/data fusion is an emerging science that incorporates elements of physics, engineering, mathematical physics, and computational science, the International Society for Information Fusion (ISIF) was created in 1999, with a constitution approved in April 2000.

For ISIF, information fusion encompasses the theory, techniques and tools conceived and employed for exploiting the synergy in the information acquired from multiple sources (sensors, databases, information gathered by humans, etc.), such that the resulting decision or action is in some sense better (qualitatively or quantitatively, in terms of accuracy, robustness, etc.) than would be possible if any of these sources were used individually without such synergy exploitation. In doing so, events, activities and movements will be correlated and analyzed as they occur in time and space, to determine the location, identity and status of individual objects (equipment and units), to assess the situation, to qualitatively and quantitatively determine threats, and to detect patterns in activity that reveal intent or capability. Specific technologies are required to refine, direct and manage information fusion capabilities.

The ISIF web site at http://www.inforfusion.org contains much of the crucial documentation in the whole domain. The results contained in this series of reports was presented in part at the first nine ISIF-sponsored FUSION conferences in

2006: Florence, Italy, at http://www.fusion2006.org

2005: Philadelphia, Pennsylvania, USA, at http://www.fusion2005.org/

2004: Stockholm, Sweden, at http://www.fusion2004.org/

2003: Cairns, Queensland, Australia at http://fusion2003.ee.mu.oz.au/

2002: Annapolis, Maryland, USA at http://www.inforfusion.org/Fusion_2002_Website/index.htm

2001: Montréal, Quebec, Canada at http://www.crm.umontreal.ca/fusion/, with both Lockheed Martin Canada and DRDC-V as sponsors

2000: Paris, France, at http://www.onera.fr/fusion2000/

1999: Sunnyvale, California, USA, at http://www.inforfusion.org/fusion99/, during which the concept of ISIF first emerged

1998: Las Vegas, Nevada, USA, at http://www.inforfusion.org/fusion98/

The ISIF community is also served by *Information Fusion*, a journal published by Elsevier (see http://www.elsevier.com/wps/find/journaldescription.cws_home/620862/description for more information) and will soon have an on-line journal of its own: the Journal of Advances in Information Fusion (JAIF).

## Specialized sources of documentation

This report has summarized the deliverables of the “Demonstration of Multi-Platform Data Fusion Between Halifax Class Frigate and an Airborne Collaborative Platform” (DFCP) contract No. W2207-E1V01 performed at Lockheed Martin Canada. There were four deliverables, listed below:

LM Canada Doc. No. 990001623, (2001), Demonstration of the Baseline System for Collaborating HALIFAX Class and Aurora Data Fusion Subsystems, Rev. 1, 6 December 2001

LM Canada Doc. No. 990001657, (2002), Demonstration of the Enhanced Data Fusion, Imaging and Simulation Capabilities within the collaborating HALIFAX Class and Aurora Data Fusion Subsystems, Rev. 1, 4 December 2002

LM Canada Doc. No. 990001678, (2003), Demonstration of the System Refinements within the collaborating HALIFAX Class and Aurora Data Fusion Subsystems, Rev. 1, 17 June 2003

LM Canada Doc. No. 990001682, (2003), Final Technical Report, Rev. 1, 30 June 2003

For the additional documentation specifically needed for this report, the References section contains the complete list.

This page intentionally left blank.

# Distribution list

**INTERNAL DISTRIBUTION**

**DRDC Valcartier TR 2004-284**

1 - Director General

3 - Document Library

1 - Head/DSS

1 - Head/IKM

1 - Head/SOp

1 - P. Valin

1 - A. Jouan (author)

1 - R. Breton

1 - J.M.J. Roy

1 - S. Paradis

1 - A. Guitouni

1 - LCdr E. Woodliffe

1 - Maj G. Clairoux

1 - Maj M. Gareau

1 - Maj B. Deschênes

1 - LCdr É. Tremblay

1 - P. Maupin

1 - A.-L. Jousselme

1 - F. Rhéaume

1 - A. Benaskeur

1 - M. Allouche

1 - A.C. Boury-Brisset

**EXTERNAL DISTRIBUTION**

**DRDC Valcartier TM 2004-284**

1 - Director Research and Development Knowledge and Information Management (PDF file)

1 - 3d Thrust Leader: Doreen Dyck, DRDC Ottawa

3 - Director CORA

101 Colonel By Drive, Ottawa, ON, K1A0K2

4 - Director General Joint Force Development

101 Colonel By Drive, Ottawa, ON

2 - Director Science and Technology (C4ISR)

Constitution Building, 305 Rideau St., Ottawa, ON

2 - Director Science and Technology (Air)

Constitution Building, 305 Rideau St., Ottawa, ON

2 - Director Science and Technology (Land)

Constitution Building, 305 Rideau St., Ottawa, ON

2 - Director Science and Technology (Maritime)

Constitution Building, 305 Rideau St., Ottawa, ON

2 - Director Science and Technology (Human Performance)

Constitution Building, 305 Rideau St., Ottawa, ON

2 - Director Maritime Requirements Sea

Louis St.Laurent Bldg, 555 Boul. de la Carrière, Gatineau, QC

1 - Director Maritime Requirements Sea 4

Louis St.Laurent Bldg, 555 Boul. de la Carrière, Gatineau, QC

1 - Director Maritime Requirements Sea 6

Louis St.Laurent Bldg, 555 Boul. de la Carrière, Gatineau, QC

1 - Director Maritime Requirements Sea 6-2

Louis St.Laurent Bldg, 555 Boul. de la Carrière, Gatineau, QC

2 - Director Aerospace Requirements

101 Colonel By Drive, Ottawa, ON

**EXTERNAL DISTRIBUTION (cont'd)**

**DRDC Valcartier TM 2004-284**

1 - Director Aerospace Requirements 4

101 Colonel By Drive, Ottawa, ON

1 - Director Aerospace Requirements 3

101 Colonel By Drive, Ottawa, ON, K1A0K2

2 - Director Maritime Ship Support

Louis St.Laurent Bldg, 555 Boul. de la Carrière, Gatineau, QC

2 - Director Maritime Ship Support 6

Louis St.Laurent Bldg, 555 Boul. de la Carrière, Gatineau, QC

2 - Director Maritime Ship Support 8

Louis St.Laurent Bldg, 555 Boul. de la Carrière, Gatineau, QC

1 - Director Science and Technology Air - 5 (3d)

Constitution Building, 305 Rideau St., Ottawa, ON

1 - Director Science and Technology Maritime - 2 (1b)

Constitution Building, 305 Rideau St., Ottawa, ON

1 - Director Science and Technology Maritime - 3 (1a)

Constitution Building, 305 Rideau St., Ottawa, ON

1 - Director Science and Technology Maritime - 5 (1c)

Constitution Building, 305 Rideau St., Ottawa, ON

1 - Director Science and Technology Land - 2

Constitution Building, 305 Rideau St., Ottawa, ON

2 - Director Land Requirements

Louis St.Laurent Bldg, 555 Boul. de la Carrière, Gatineau, QC

1 - Canadian Forces Experimentation Centre

Shirley's Bay Campus, 3701 Carling Ave., Ottawa, ON

2 - Defence R&D Canada – Toronto:

Attn: J. Hollands

K. Hendy

**EXTERNAL DISTRIBUTION (cont'd)**

**DRDC Valcartier TM 2004-284**

1 - Defence R&D Canada – Ottawa:

Attn: A. Damini

4 - Defence R&D Canada –Atlantic:

Attn: J.S. Kennedy

B. Chalmers

B. McArthur

M. Hazen

2 - PMO Maritime Helicopter Program

Attn: P. Labrosse

Col. W.O. Istchenko

1 - PMO Halifax Modernized Command and Control System

Attn: DMSS 8

1 - PMO Aurora Incremental Modernization Program

Attn: G.B. Lewis, DAEPMM 5

1 - Canadian Forces Command and Staff College
Toronto
Attn: Commanding Officer

1 - Canadian Forces Maritime Warfare School
CFB Halifax
Halifax, Nova Scotia
Attn: Commanding Officer

2 - Canadian Forces Maritime Warfare Centre
CFB Halifax, NS
Attn: TAC AAW
OIC Modelling and Simulation

2 - Canadian Forces Naval Operations School
CFB Halifax, NS
Attn: Tactics
CT AWW

1 - Canadian Forces Naval Engineering School
CFB Halifax, NS
Attn: CSST

1 - Operational Requirements Analysis Cell

CFB Halifax
Attn: Commanding Officer

1 - Canadian Forces Fleet School
CFB Esquimalt, BC
Attn: Commanding Officer/WTD

1 - Operational Requirements Analysis Cell
CFB Esquimalt, BC
Attn: Commanding Officer

**DOCUMENT CONTROL DATA**

| | |
|---|---|
| 1. ORIGINATOR (name and address)<br>Defence R&D Canada Valcartier<br>2459 Pie-XI Blvd. North<br>Québec, QC<br>G3J 1X5 | 2. SECURITY CLASSIFICATION<br>(Including special warning terms if applicable)<br>Unclassified |

3. TITLE (Its classification should be indicated by the appropriate abbreviation (S, C, R or U)
Concepts of Data/Information Fusion for Naval C2 and Airborne ISR platforms (U)

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj. John E.)
Jouan, A., Valin, P., Bossé, E.

| | | |
|---|---|---|
| 5. DATE OF PUBLICATION (month and year)<br>November 2006 | 6a. NO. OF PAGES<br>117 | 6b .NO. OF REFERENCES<br>36 |

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. Give the inclusive dates when a specific reporting period is covered.)
Technical Report

8. SPONSORING ACTIVITY (name and address)

| | |
|---|---|
| 9a. PROJECT OR GRANT NO. (Please specify whether project or grant)<br>13DV | 9b. CONTRACT NO. |
| 10a. ORIGINATOR'S DOCUMENT NUMBER<br>TR 2004-284 | 10b. OTHER DOCUMENT NOS<br>N/A |

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

- [x] Unlimited distribution
- [ ] Restricted to contractors in approved countries (specify)
- [ ] Restricted to Canadian contractors (with need-to-know)
- [ ] Restricted to Government (with need-to-know)
- [ ] Restricted to Defense departments
- [ ] Others

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

The main objective of this report is to analyze methods, techniques, algorithms, rule-based communication protocols and infrastructures needed to establish a Maritime Tactical Picture (MTP). An MTP is, by definition, the combination of the Local Area Picture (LAP) seen by a unit (which may be part of a Task Force) using its own sensors and a Wide Area Picture (WAP) using information, not controlled by the Task Force, provided by High Frequency (HF) or Ultra High Frequency (UHF) radios or satellites. For that purpose a test-bed was developed for testing and benchmarking all these above-mentioned elements. This test-bed provides the tools needed to study what information should be communicated, when and how, and the impact it has on the joint situational awareness.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Information fusion, data fusion, CP-140 Aurora, HALIFAX frigate, Knowledge Based System, multiple blackboards, decentralized fusion, tracklets, inverse Kalman filter, inverse information filter, covariance intersection, information prioritization, track quality, track reporting, local track database, global track database, simulation environments, demonstration of algorithms, FLIR classifiers, fusion of classifiers.